AIAA Flight Simulation Technologies Conference

A
COLLECTION
OF
TECHNICAL
PAPERS

San Diego, CA
July 29-31, 1996

AIAA

For permission to copy or republish, contact:
American Institute of Aeronautics and Astronautics
1801 Alexander Bell Drive
Suite 500
Reston, VA 22091

A COLLECTION OF TECHNICAL PAPERS

AIAA Flight Simulation Technologies Conference

July 29-31, 1996/San Diego, CA

ISBN
1-56347-216-3

Printed in the U.S.A.

AIAA Flight Simulation Technologies Conference
July 29-31, 1996
San Diego, CA

Table of Contents

Rotorcraft and Air Cushion Vehicle Dynamics & Modeling

Civil Aviation

Avionics, Weapons and Engagement Modeling & Simulation

NA - Not Available WD - Withdrawn

Space Systems I

Aircraft Dynamics, Modeling & Performance

NA - Not Available **WD - Withdrawn**

Hardware/Software Technologies

NA - Not Available **WD - Withdrawn**

Advanced Concepts and Applications

Training, Testing, Verification & Validation

Space Systems II

Networked Technologies & Applications

NA - Not Available **WD - Withdrawn**

Low Cost Simulation, Virtual Reality, Database Development & Atmospheric Modeling

Human Factors & Pilot Cueing

NA - Not Available **WD - Withdrawn**

SOFTWARE DESIGN FOR A RECONFIGURABLE SIMULATOR

P.C.A. van Gool *

Faculty of Aerospace Engineering
Delft University of Technology, the Netherlands
E-mail: P.vanGool@LR.TUDelft.NL

Abstract

This paper describes the software design for the SIMONA Research Simulator (SRS). The main components and subsystems of the SRS are discussed, including their interconnections. Some of the techniques used to obtain a deterministic communication scheme, while using relatively inexpensive common-of-the-shelf (COTS) hardware, will also be brought forward.

Through discussions with future users the requirements of the system have been defined. From these requirements it is apparent that the simulation process as a whole can be split up into several modes. These modes will lead to a state transition diagram which can later be used when implementing the system. Each mode will be discussed and each transition will be clarified.

The paper will then focus on the software running in the simulation host computer. In order to satisfy the aforementioned requirements, the software in the host is split up into several modules, each satisfying one or more of the system requirements. Each of the identified modules or 'managers' will be discussed. Besides their functionality, some of the implementation issues will also be discussed.

Introduction

The Delft University of Technology is developing the International Centre for Research in Simulation, Motion and Navigation Technologies (SIMONA), with the intention to carry out significant fundamental research into simulation techniques and standards, as well as vehicle control. Its programs operate within the DUT under the official SIMONA Inter-Faculty Working Group in which the Faculty of Aerospace Engineering cooperates with the Faculty of Mechanical and Marine Engineering, and the Faculty of Electrical Engineering. Central to SIMONA is a multi-purpose re-configurable simulator (see Fig. 1). This device will allow researchers to perform experiments on, for example, human perception, mathematical modelling, flight control system design, and motion system control. These requirements in turn demand a flexible software structure[1].

A primary technological aim of SIMONA is to conduct *fundamental* research into simulation techniques and to thereby improve the design of simulator systems. SIMONA programs will devote attention to motion systems, hydraulics and control, and navigation systems modelling, as applied to simulation. The second major scientific goal is to provide better insights into human-vehicle interfaces for both flight and non-flight vehicles.

One of the key requirements for the system is flexibility, in order to allow the system to be con-

*Research Assistant, Ph.D. Candidate

figured rapidly and executed with deterministic results. This should be applicable for a variety of simulation experiments. A second priority, one which should never be overlooked, is safety. It is necessary to always prevent personal injury or damage to equipment. The software system will therefore have to allow a high level of flexibility while simultaneously monitoring the safety of the system. This means that the software must not only guard system integrity to the outside world (the experiment leader), but also among the several software modules or processes within the system itself. These requirements will reflect upon the design of the system.

One of the responsibilities of the host is to calculate the response of the vehicle to the pilot's inputs, atmospheric conditions and the environment in general. Because the SIMONA Research Simulator will have to accommodate several types of vehicles, and vehicle software modules should also be re-usable, a redesign will be made of the mathematical models for the vehicles. Some issues on this matter will be discussed.

Simulator Subsystems

In this section the various simulator subsystems will be discussed. Each of these subsystems performs a specific task. Generally speaking, each subsystem's task is to stimulate/fool one of the pilots senses. The SIMONA Research Simulator is comprised out of a number of subsystems. A picture of the complete system can be seen in Fig. 2.

Simulation Host

The simulator host computer is the device which executes the simulation software and controls the connection and communication to the simulation subsystems. A computer that supports one or more simulation applications. All host computers participating in a simulation exercise are connected by a common network.

Its functions will include the following.

- During non real-time operations the simulation host will perform as file server for the subsystems.

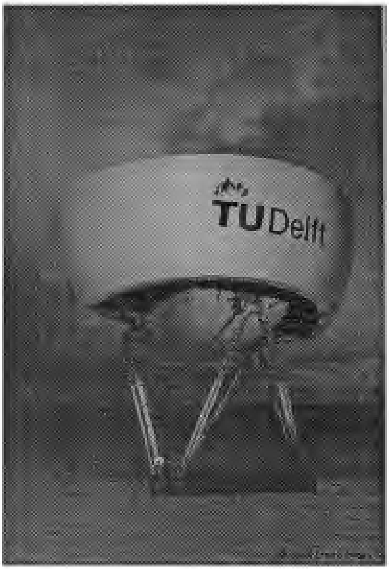


Figure 1: The SIMONA Research Simulator.

- The simulation host must calculate the response of the vehicle being simulated. This response will include the effects of pilot input, atmospheric conditions, system malfunctions and interactions with other entities.
- The simulation host will keep track of all other entities that are active in the current simulation.
- The simulation host will contain a model of the present state of the atmosphere.

Hardware-in-the-loop

One of the subsystems is hardware-in-the-loop. This makes it possible to include hardware that is

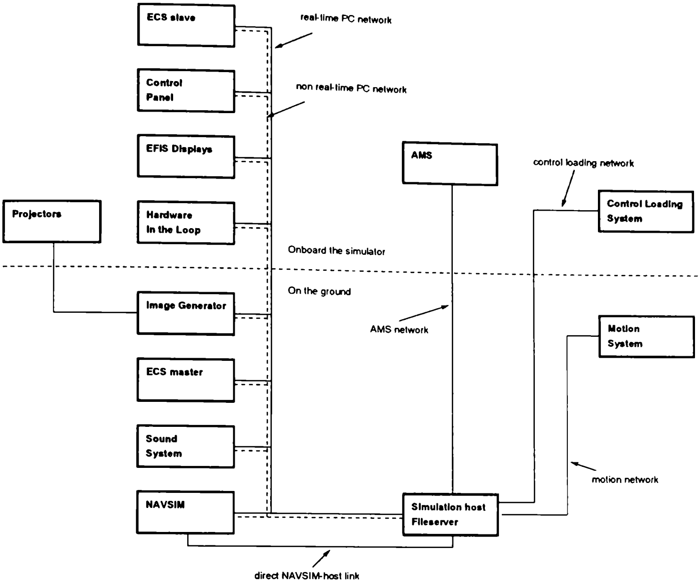


Figure 2: The SRS Subsystems.

actually used in real aircraft. It should, for instance, be possible to include the real flight management computer in the real-time loop. Data traffic between the real-time host and that hardware should comply to the ARINC-429, ARINC-629 and MIL-1553 standard. The connection of the hardware-in-the-loop to the simulation host will be through a PC that will be connected to the real-time network.

Experiment Control Station

Two ECS's are to be used, the first console is located in the simulator and the second in the control room. This will enable the experiment leader to control the simulation in all phases of the simulation except the initialisation phase. For a further discussion of the possible phases or states of the simulation, see page 6.

The ECS will allow the experiment leader to dynamically change the simulation environment by, for instance, changing the atmospheric conditions. It will also allow him to specify data, that will be generated during the simulation, to be logged, to make snapshots, movies, etc. The actual logging and storage of snapshots will be done by the host, but the ECS will give the experiment leader the controls to enable them.

It will also allow "people" outside the simulator to keep track of what is happening inside the simulator because the ECS in the control room will have the same output as the one inside the simulator.

The interaction of the ECS with the simulation will be through a graphical user-friendly interface. The ECS will contain several displays and controls, which provide the experiment leader information about and control over the simulation before and during a simulation session. The control devices will be designed in such a way as to reduce the workload for the experiment leader, to provide clear displays, and simplify the job of initialising, monitoring and changing the experiment. It will support, as a minimum:

- A means to monitor and control the simulation, providing subsystem status feedback and supplying information about subsystem status.
- A means to specify a scenario for an experiment session.
- A means to enable the experiment leader to control all required system variables and insert abnormal or emergency conditions into the aircraft systems.
- A means for the experiment leader to function as a remote pilot.

Control Loading Subsystem

The control loading subsystem takes care of the pilot controls in the simulator. It is used to give the pilot feedback of his actions by simulating the control forces a pilot would feel in the actual aircraft. It is also used to measure the pilot's control deflections.

These devices can be either electrically or hydraulically loaded devices. The SRS will provide the pilot with conventional aircraft controls, with column or side-stick, and helicopter controls.

Motion Subsystem

The motion subsystem will control the movement of the motion platform. This subsystem requires from the host the desired platform acceleration, velocity and position in order to control the platform.

This subsystem will act as a slave to the host. All motion is specified by the host. The motion computer will receive setpoints from the host to control the platform.

Software for this system is developed at the Department of Mechanical Engineering. It is developed using the Matlab/Simulink environment after which code is automatically generated and downloaded into the DSPs.

Sound Subsystem

Till now the use of sound systems in commercial simulators has been very limited. The SRS will have a much more advanced sound system than the ones that are currently in use. The reasons for increased activity in that area are:

- A more realistic (3D) representation of aural cues during a simulation will increase the realism of the entire simulation.
- New research areas will become available. So far aural cues have been used to inform the pilot about problems and dangerous situations. These conventional aural cues only tell the pilot there is a danger, for example. In the future, research has to be done into spatial aural cues, which will also present the pilot with the position of the problem.

Visual Subsystem

The visual subsystem will provide the pilot with the outside view. Establishment of the location in the world, will be done through the ECS. The image generator is likely to be off-the-shelf, but the display system will be purpose built.

EFIS Subsystems

In the simulator, the flight instruments will be presented to the pilot through simulated EFIS displays. Which instruments will have to be presented on what display, will have to be decided during in configuration phase. For a further discussion of the possible phases or states the simulation could be in, see page 6. High-performance graphics hardware will be used to generate rapidly two- and three-dimensional display formats for evaluation in a full glass cockpit.

Flight Control Panels

Not all pilot input will enter the host via the control loading system. Some discrete inputs will come to the host through the flight control panel and additional levers or switches in the cockpit. Examples of these kinds of input are those from the landing gear lever, and from the mode control panel. Aircraft systems like a simulated FMS will be connected to the system through this subsystem.

NAVSIM

The NAVSIM subsystem will be responsible for the simulation of the navigation systems onboard the aircraft. NAVSIM will be running on a separate computer. Output of the NAVSIM system will be used to drive the various EFIS subsystems. A possibility is to have NAVSIM include propagation of radio signals through the atmosphere into the dynamics of the navigation systems. Whether or not this will be included in a real-time simulation depends on the performance of the subsystems.

The main focus will be on radio navigation aids. To be able to function, NAVSIM has to use a lot of data from other simulation subsystems of SIMONA. To be able to simulate all aspects of radio-navigational aids, ionospheric and tropospheric conditions (i.e. atmospheric conditions further away from the aircraft) also have to be available.

Acceleration Measurement System

The Acceleration Measurement System (AMS) is a system that can be used to measure linear and angular accelerations onboard the simulator. Such a system can be used to determine the latency of the simulator or measure vibrations of the visual display.

Subsystem Connections

The network that will connect the simulator subsystems will consist of several Ethernet branches. A branch may form a point-to-point link between two subsystems or a link between more than two subsystems. The host computer is always one of the subsystems on a link.

The control loading subsystem and motion subsystem will each have a dedicated glass-fibre link to the host. In theory, they too could be combined on an Ethernet link with other subsystems. Because of hardware choices and safety arguments, both subsystems will have a dedicated connection to the host computer.

As can be seen in Fig. 2 subsystems on the PC network have two network connections, a real-time one and a non real-time one. Diskless subsystems will boot through the non real-time network and all subsystems will receive their simulation programs through that network also. This will be done using standard NFS facilities. This setup will allow subsystems to crash without bringing down the entire system. They will try to reboot through the non real-time network which will not interfere with the deterministic communication scheme on the real-time network. Another use for the non real-time network is for safety. If a system is malfunctioning, the host computer can communicate with that subsystem through the non real-time network and allow the real-time communication to continue.

Ethernet is a bus network. This means that all processors connected to the network share the same media. Processors that want access to the media must contend. Ethernet arbitrates media contention in hardware using the Carrier Sense Multiple Access with Collision Detection (CSMA-CD) protocol[2]. This protocol can cause unbounded access delay to the media. The Ethernet adapters use a binary exponential backoff mechanism to (re)gain access to the network after a collision.

Consequently, the CSMA-CD protocol is considered unsuitable for use in a real-time environment in the sense that it does not guarantee bounded media access delays. However, Ethernet can still be used in a real-time environment if the media contention is solved in higher protocol layers, that are implemented in software. By solving the contention problem at a higher level, the hardware-implemented binary exponential backoff algorithm is never activated.

A protocol has to be used such that bounded network communication time (for bounded message length) is achieved. One such a protocol is Timed Master-Slave (TMS). The protocol must take into account safety, must be scalable and predictable.

The simulation host computer is designated as master. All other subsystems become slave. They may not access the network unless explicitly commanded by the master. The master commands a slave, that is allowed access to the bus for a fixed time slot starting immediately after reception of the command. The master waits until the slave's time slot is finished and the process is repeated. If the slave does not respond within a certain time limit, it is considered to have failed. It is then the host's responsibility to activate the appropriate safety policy.

After the slave receives the host's command, it must reply immediately. It does not have time to calculate the reply, or gather it from its I/O devices. This, in effect, can be viewed as a task that sends a message at the end of its execution, but has a delayed access to the communication media.

Simulation Modes

Through analysing the requirements the users place upon the system[1], it was apparent the system as a whole has to perform/allow certain tasks in different operating states. The analysis shows that the simulator operation can be divided up into several states. These states are shown in Fig. 3.

The simulation process can run in two modes: real-time and non real-time. The non real-time mode will be active during the initial startup phase of all hardware. During this time EFIS display geometry and the requirement to join a DIS session will have to be specified. After the initialization phase is completed, the real-time mode can be enabled.

This state transition diagram can be considered a skeleton for all software in all simulator subsystems.

In this section the modes of the simulation process will be described. The relations between the modes is illustrated in Fig. 3.

The INIT Mode

When the simulation session is in the INIT mode, it means that it has just been started. The communication with the subsystems has to be initialised and the subsystems will need to boot and load their applications.

The ECS display will also be initialised and it should then be possible to control the entire simulation session from the ECS display.

The CONFIG Mode

In this mode the subsystems have initialised successfully and are now waiting for information concerning their configuration. The experiment leader can now configure the virtual world, EFIS display geometry, aircraft type to be simulated, etc. It should also be possible to load an entire simulation configuration from disk to expedite this process.

The START Mode

The network is now in real-time mode. In the START mode, the initialised system is already prepared for actual simulation. From START the simulation can be terminated (SHUTDOWN), the configuration can be changed (CONFIG) or a simulation session can be started. The bridge to the simulator is then removed, and the motion platform is directed to its neutral position. In order to reconfigure the system, it will have to leave real-time mode.

The HOLD Mode

HOLD mode performs an important role in the simulation process. It is from this mode that many of the remaining modes can be activated: RUN, RESET, REPLAY and back to START. Back to START means lowering the platform and extending the bridge.

During HOLD it is still possible for the experiment leader to change the world and atmospheric conditions. It is no longer possible to change the EFIS displays configuration or aircraft type.

The RUN Mode

In this mode the cockpit controls, switches etc. are sampled at 60/120Hz. This data is then transmitted to the host through the network. The vehicle simulation will determined the response of the vehicle to the input. With that information the subsystems can then be updated. This is done under control of the host. The host will specify which subsystem is allowed to transmit its data.

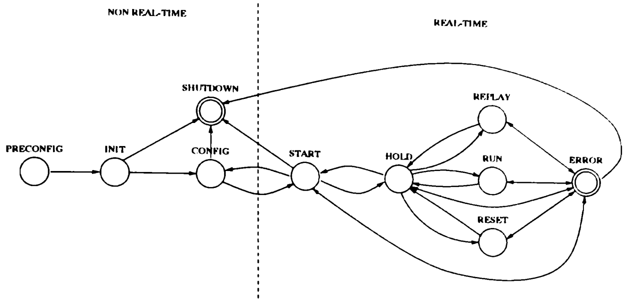


Figure 3: Simulation States.

In case of errors, the process will proceed to the ERROR mode. The simulation can also be interrupted by going back to the HOLD mode. This can be achieved by the experiment leader by using the ECS or by the pilot by pressing the hold button in the cockpit.

The REPLAY Mode

If during the simulation session or during a previous simulation session movie data has been saved, this can be used to replay that simulation. By proceeding from the REPLAY mode to the HOLD mode and then to the RUN mode, it is possible to, for instance, practise an approach several times. This can also be achieved by using snapshots.

The RESET Mode

When the pilot or experiment leader wishes to return to the position the simulation started, the RESET mode must be activated. The motion platform will also return to its initial position.

The SHUTDOWN Mode

In this mode, activated when the simulation session is over, all subsystems will be shutdown and all communication with the subsystems will be terminated. Data will be saved to disk and the simulation executive will be removed from memory.

The ERROR Mode

When during any mode a serious error occurs in one or more subsystems, the ERROR mode will be activated and the entire simulation systems will be shutdown in an orderly fashion.

If the error is considered to be non-critical the experiment leader will be notified of the error and he can decide whether or not to terminate the session.

Host Computer

The SIMONA Research Simulator (SRS) will be a reconfigurable simulator in the sense that hardware as well as software can be changed to configure a new simulator. This requirement dictates that both hardware and software be modular in order to minimise the time needed to change from one configuration to the next. Rapid configuration changes should be possible by simply changing configuration files and restarting the system.

In this section the basic layout of the simulation host software will be given[3]. The layout of the software and the connections between the various objects can be seen in Fig. 4. A general description of these objects will be given to give an idea of the concepts behind the layout.

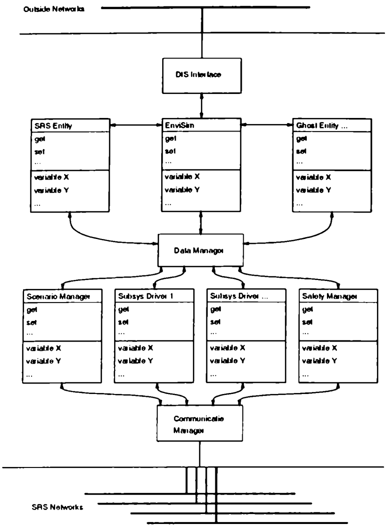


Figure 4: SRS host software.

Distributed Interactive Simulation

The SRS will allow the experiment leader to have his subject fly in an arbitrary world. In this world, the subject is not necessarily alone. There can be ghost aircraft (aircraft flying according to a predetermined script) or aircraft controlled by other humans. These humans can also be flying a simulator or can be flying using a desktop workstation. Communication with the other humanly controlled aircraft will have to comply to the DIS protocol.

The DIS standard requires the simulation host to keep track of all entities involved in the simulation run. It will have to calculate the positions of the entities by using dead-reckoning. If the entities not controlled by the host start to deviate too much from the dead-reckoned position, the host will receive a new update of the entity's state.

In the same way the host will have to send out the new state of the entity it is simulating if the deviation from the dead-reckoned position becomes too big. This means that the simulation host will have to calculate the real and the dead-reckoned position of the entities it controlles during the simulation run.

The process that will take care of the Distributed Interactive Simulation (DIS) communication is denoted in Fig. 4 by DIS. It will have the ability to send and receive (write and read) DIS messages, called protocol data units or PDUs. Before it can do this it has to open the network connection and after a simulation it also has to be able to close it. Finally, the process must possess the ability to filter out certain PDU types.

The ability to filter out certain PDU types can be used to decrease the number of DIS messages that has to be interpreted by EnviSim (see further on).

Environment Simulation

For the SIMONA Research Simulator, it is important that not only the entity simulation will use the environment simulation but also the NAVSIM subsystem. NAVSIM is involved with the modelling of radio navigation and as such is interested in the propagation of radio waves through the atmosphere. While for the entity simulation it is usually sufficient to know, for example, the air temperature near the centre of gravity of the entity, for radio navigation simulation it is important to know the air temperature in a wide region around the entity.

Another important aspect of environment modelling is that a coherent environment should be presented to all entities that are a part of the simulation exercise. Two entities in close vicinity of each other should, for instance, experience the same turbulence intensity.

This leads to the conclusion that the environment simulation is not merely a subroutine in the simulation process of a separate entity, but a separate process by itself.

For the SIMONA Research Simulator, the environment simulation (EnviSim) will have to supply the entities with information about, for example, the atmosphere and terrain. By centralising the management of these two, there will exist a coherent environment between the entities involved in the simulation.

When the simulation application is part of a DIS exercise, EnviSim will also take care of DIS communication. Because EnviSim knows which entities are part of the simulation, where they are and what their state is, it is best to leave this communication to EnviSim. The alternative would be to incorporate this functionality into the entity models. It is thought that that would further complicate network communications.

Entity Simulation

One of the requirements is that the host computer must be able to simulate multiple entities and not just one. Why is that? We want multiple entities because, like in our real world, we want to interact with other entities. When driving a car, we are rarely alone on the road. In fact, the road itself could be considered an entity.

Control of these entities can come from several sources. One of the entities will be controlled by the person in the SIMONA Research Simulator (SRS), of course. This entity will be called the SRS entity to distinguish it from other entities. Other entities may be controlled by scripts running on the same host or on a completely different computer. Finally, it will also be possible that entities are controlled by persons in another simulator or by persons using low-end desktop computers.

All entities controlled by other computers or other persons than the one in the SRS will communicate with the host by using the standard Distributed Interactive Simulation (DIS) protocols. In order to comply with the DIS protocols, the host computer will have to simulate the dynamics of those entities. There is however a difference in the way the dynamics for these entities will be calculated. Dead reckoning will be used. Concluding, the host computer will have to keep track of all the entities involved in the simulation session.

The sort of entities that can be expected to be involved in a simulation varies enormously. For example, a cloud too can be regarded as an entity that moves around in the environment as a result of its own dynamics and the interaction of the entity with the environment.

The entities shown in Fig. 4 are the entities for which the host is responsible and for which the host will calculate the dynamics. These are therefore not the DIS entities. They will be dealt with inside EnviSim.

The software for entity simulation will have to be modular. This way it should be possible to reuse some of the software. The equations of motion of an entity could be considered as a standard building block when developing a new simulation. It makes no sense to program them again and again. An entity simulation model can also be split up into several parts.

SRS Entity Simulation

As was mentioned earlier there is one special entity to be simulated. That is the SRS entity. The SRS entity is the entity that will be used by the SIMONA Research Simulator. One of the differences between the SRS entity and the other entities is that the state of the SRS entity will be used to determine the states of most of the subsystems. The state for these subsystems can be determined after some transformation.

The transformation between the SRS entity state and the motion platform state is modelled in the motion drive laws. This process is also shown in Fig. 4 as a 'subsystem driver'. Whether or not a subsystem needs a transformation like that depends on the subsystem in question.

Data Manager

The data manager is the process that will make the connection between the entity simulations and the communication facilities of the host to the SRS networks. Before the simulation actually begins, the data manager determines whether or not the requested configuration will make a valid simulation. It will do this by examining the input and output data of all the requested configurations of the subsystems and the input and output of the requested simulation modules in the host. A match must exists between them. When an engine instrument is selected that displays the data of 4 engines and the SRS entity is selected to be an aircraft with 2 engines there is a mismatch of data. The data manager will recognise this and report it as an error.

Once the data manager has determined that the requested simulation configuration is valid, it can determine what communication is needed between the modules and the subsystems. It knows where to get the data through configuration files that tell it who has what data. It can then set up a communication sequence.

Another important responsibility of the data manager is to supply data to the scenario manager during a simulation. Because it must be possible to program or change scenarios during a simulation, the communication between the data manager and the scenario manager is dynamic, meaning it can change during a simulation.

Scenario Manager

The scenario manager is used to execute scenario's during a simulation. A scenario is a small program that can insert events into the simulation application. The scenario manager communicates with the data manager. It tells the data manager which data it needs to execute its scenario. If that data is not present in the simulation application, the data manager will tell the scenario manager the scenario is invalid for this simulation. The data the scenario manager needs can be used for input to and output from the scenario manager. There is of course a safety issue here which will need to be addressed.

Communication Manager

The communication manager will send data to and receive data from the SRS networks. Depending on the configuration of subsystems, the communication manager will retrieve input and output frequencies, data descriptions, access modes, etc. from configuration files.

The communication manager will keep track of all network communication. If a subsystem does not respond within a certain amount of time or sends back an invalid status or a failure status, the communication manager will notify the subsystem driver which will in turn notify the data manager. Note that in this way the data manager is completely unaware of the communication manager. Unless notified to stop, the communication manager will then continue to perform its functions.

Safety Manager

The function of the safety manager is to decide what to do if an error occurs in one of the subsystems. This possible failure of the subsystem will be registered by the communication manager which will notify the data manager through the subsystem driver. The safety manager will have to know the current hardware configuration of the system and the accompanying safety policies.

If case of an significant subsystem failure, the safety manager will instruct the communication manager (through the data manager and subsystem driver) to inform the experiment control station of the upcoming shutdown and will then inform the subsystems to shut down.

Mathematical Models

In a reconfigurable simulator like the SRS it is also desirable that the mathematical models for the entities are reconfigurable. This way when a new entity has to be simulated building blocks from other entity simulation can be used.

Also the mathematical models shoulu be developed with the physical system in mind. The advantage of such an approach is that a complex physical system can be modelled by simply assembling its component parts. This approach is similar to the principal of functional decomposition/recomposition and to the spatial discretisation methods of Finite Elements. As in FEM, the system representation, is built up by assembling primitives with known behaviour in a framework mimicking the real system. Some of the advantages of using this approach are:

- model transparency,
- re-usability,
- reconfigurability.

The direct correspondence of the object model to the real system enhances understanding and should lead to rapid model development cycles. The modelling approach draws heavily on a set of primitives which are re-usable across different models. The physical nature of the models, permits a developer to make straightforward modifications to the model, based on similar changes to the real vehicle. Additionally, by suitably varying the functional attributes of the objects, the model can be reconfigured with ease.

An Example

In this subsection an example of the above discussed principles will be given. For this example the propulsion system of an aircraft will be used. The idea is that somebody who wants to define a propulsion system does not want to be involved in the modelling of the actual engine dynamics. All that person wants to do is use a library of engine models, pick some out and build a propulsion system with them.

Fig. 5 is an example of a datafile defining the configuration of a Cessna Citation 500 propulsion system. Such a setup should be used for other entity submodels too. The process in charge of calculating the propulsion forces can use this definition file to calculate the total forces and moments that all engine contribute to the entity in the model reference frame. It will send this data to the process in charge of calculating entity body dynamics. That process will transform the forces and moments to the body axes reference frame around the centre of gravity and use that to update the entity's state.

Conclusions

In this paper the SIMONA Research Simulator system has been described. All subsystems have been identified and a short description of each has been given. The solution used to connect the subsystems through a real-time network, using off-the-shelf hardware, has been described.

The simulation process has been defined globally as a finite state machine. Each mode has been clarified. By clearly defining the state transition requirements (pre- and post-conditions) the basis for all subsystem software is defined.

```
# Begin of propulsion.def
#
# This definition file contains:
# Citation propulsion system
#
# The number of engines requested
NUM'ENGINES 2
# For each of the engines specify type, location
# w.r.t. model reference point [m], tilt angle
# [rad] and toe angle [rad].
#
# num.   type     x     y     z   tilt   toe
#-------------------------------------------------
ENGINE1 JT15D   0.000  1.312  0.000  0.000  0.000
ENGINE2 JT15D   0.000 -1.312  0.000  0.000  0.000
# End of propulsion.def
```

Figure 5: Definition of a propulsion system.

The simulation host computer is also identified and investigated. Its desired functionality is analysed and a design is made of the several objects that will run in the host computer. Each of these objects (or managers) is briefly described.

The purpose of this paper has been to show that, although a 6-DOF flight simulator is a complex system to develop and control, when using a systematic approach (requirements, analysis, design and implementation) even this system can be developed in a manageable fashion.

References

[1] P. van Gool, "SIMONA Research Simulator Software Requirements Document," tech. rep., SIMONA, 1995.

[2] T. Lamerigts, "Real-Time Communication in SIMONA Research Simulator," Master's thesis, Faculty of Electrical Engineering, Delft University of Technology, 1995.

[3] P. van Gool, "Simulation Host Software Requirements Document," tech. rep., SIMONA, 1995.

AIAA-96-3471-CP

SCALEABLE DESIGN FOR RECONFIGURABLE MANNED INTERACTIVE CREWSTATIONS

Bill Beavin
McDonnell Douglas Corporation
St. Louis, Missouri

Abstract

McDonnell Douglas Aerospace (MDA) has developed a modeling and simulation approach which allows scaleable fidelity of interaction with an aircraft simulation, while not sacrificing internal software model fidelity. This design allows many of the same assets to be reused for a wide range of simulated aircraft. A key design philosophy is the abstraction of the computational software models from the crewstation display and interaction means. This abstraction allows for seamless scaleability of software models, graphical displays, and cockpit interaction equipment. Development of new aircraft concepts and modifications to current aircraft are assisted through RMICS development. The system is Distributed Interactive Simulation (DIS) Protocol Version 2.04 compliant. It has been used to assist in the development and demonstration of the Joint Modeling And Simulation System (J-MASS) program. The mobility of the system lends itself well to conferences and trade show exhibits, as well as various demonstrations in the Pentagon, Air Force Bases, and the offices of our virtual aircraft customers. Future work includes enhanced DIS Interoperability, PC based RMICS systems, and more realistic software models. A J-MASS based Reconfigurable Cockpit is planned. Continual expansion of the RMICS aircraft repertoire is expected, along with integration with interactive warfare simulators like Interactive Warfare Simulation (IWARS).

Background

It is intriguing to consider the utility and affordability of an interactive aircraft engineering simulation which can run independently on the desks of modern day engineers, allow them to develop and test their systems independently, merge their software modules with other software modules, and easily scale up the level of fidelity to appropriately, adequately, and affordably test, demonstrate, or evaluate the aggregate desired functionality of either software or flight hardware. It is important that the increase in interactivity be seamless, and software under test be consistent, in order to minimize the effort to scale the interactivity, as well as maximize the test efficiency through the use of appropriate interaction means.

In the past, the desktop or deskside environment has been inadequate for efficiently testing many aircraft simulation functions requiring interactivity. Desktop tests were often batched, scripted, or print statement and debugger session oriented to the extent as to add layers of support software to simply test functions that could easily be observed in a more interactive environment. Often software would be developed on the desktop and checked as thoroughly as possible, and then scheduled high fidelity simulator time used to checkout the new feature. Should there be a problem, more off-line changes would be required, and more simulator time needed for checkout. The need for several levels of interactivity options was clear in order to more efficiently develop interactively intricate tasks.

As an example, a common software modification in fighter aircraft simulations consists of symbology changes on a particular display, a modification to the avionics software which drives the display, and a modification to the crewstation to interact with the symbology. Each of these three areas: the display, the avionics, and the crewstation can and should be independently tested, but a complete test requires the user to physically interact with a cockpit, send control signals to the avionics software, drive the modified symbology, observe the results, and potentially perform this test ultimately in a stressful, high workload, realistic aircraft environment. This particular example demonstrates the utility of digital tests for the three separate functional areas. In the past, the user would then test the overall functionality in a high fidelity simulator, usually along with several other modifications in order to maximize the number of tests performed during the simulator period.

It was clear in cases, like the one discussed above, that at most one or two levels of intermediate testing would have adequately exercised many basic interactivity modifications. If the display software, the avionics software, and an emulation of the actual crewstation through a keyboard and mouse were available on a simple desktop unit, a simple cockpit display button press could have been fully tested on the software engineer's desktop. Had the test required some level of dynamic hand controller input, very low cost game-like controls connected directly to the desktop unit's serial interface ports would be sufficient to test basic functionality. After these tests are completed satisfactorily, the user may then require simple tests in a more realistic and dynamic environment requiring realistic control inputs and a level of situational awareness representative of many actual flight situations.

This intermediate test environment did not previously exist, and it was towards this technology gap that the RMICS program was directed. A limited study of computational system costs and relative performance characteristics (see Figure 1) has indicated that a scaleable design can have significant impact on simulations whose computational and graphical systems are in the $100,000 or less category, where the graphics and crewstation interactivity levels increase sharply. Systems in this category are typically used for development stations, part task trainers and demonstrators, and low cost interactive aircraft simulations for use in networks with many participants.

Design Considerations

Table 1 contains the results of a limited interactivity payoff study within the "Big Picture" software development group at MDA. Informal observation and extrapolation were used to estimate average

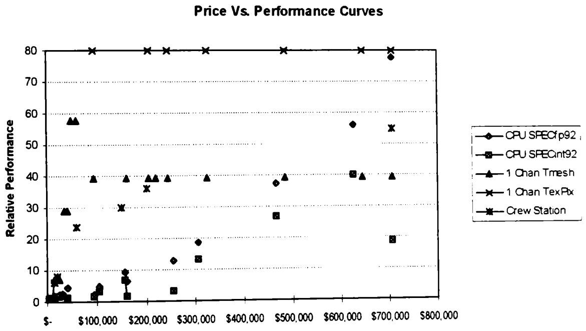


Figure 1: Relative Performance Characteristics (1)

times to perform various tasks, assuming the indicated resources were available. Table 2 applies estimated costs to the time and resources indicated in Table 1, and indicates how long it would take to recover the test equipment costs assuming optimal use. This informal study was intended to simply give rough orders of magnitude indicating interactivity payoff trends.

Development in the reconfigurable hardware and software arenas were equally important to meet the needs of software design engineers. The RMICS hardware development was directed to fill part of the interactivity level gap between the desktop and the dome. It also was to identify further needed transition configurations in both lower and higher fidelity interactivity levels, and allow the software developer to seamlessly transition between all interactivity levels.

The software appropriate for a reconfigurable simulation was also investigated, drawing upon mostly existing software and reusing it as modularly as possible. Most available software reconfiguration models were developed as part of the "Big Picture" program from 1985 through 1993, when the first RMICS was built. The "Big Picture" software was reconfigurable in nature, and had been used extensively for several research and demonstration efforts, including its airshow debut at Farnborough in 1988. Most of these software model modules were written in FORTRAN (although all display graphics code was written in "C"), obviously limiting the software reconfiguration options available, but a sufficient compromise to produce an initial functional prototype suitable for reconfigurable simulation design explorations, as well as providing a working medium fidelity aircraft simulation suitable for software development, trade show demonstrations, internal multiple-pilot simulations, and initial cockpit familiarization for pilots before entering the higher fidelity simulators.

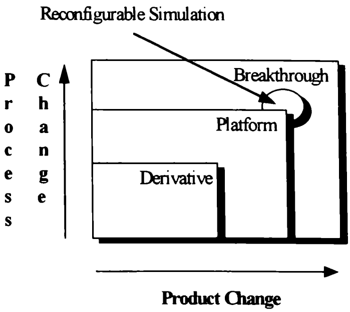


Figure 2: Aggregate Program Mapping

Using an aggregate project mapping technique[4], Figure 2 shows how reconfigurable simulations have thus far been used. Breakthrough products which identify and use new technologies as well as new processes are ideal candidates for modeling with a reconfigurable simulation.

As products and processes mature, the related products can evolve into platform products with future derivatives. It is desirable for the next generation reconfigurable simulation to be platform oriented, bringing reconfigurable simulation into the simulation mainstream.

Figure 3 shows how several recent development programs have drawn upon the RRC software, as well as using the original RMICS hardware built in 1993, and its replacement completed in 1995. Joining the original "Big Picture" Reconfigurable Cockpit users, as well as Pilot Vehicle Interface developers were T-45 Cockpit-21 designers, with plans for a future independent simulation software platform.

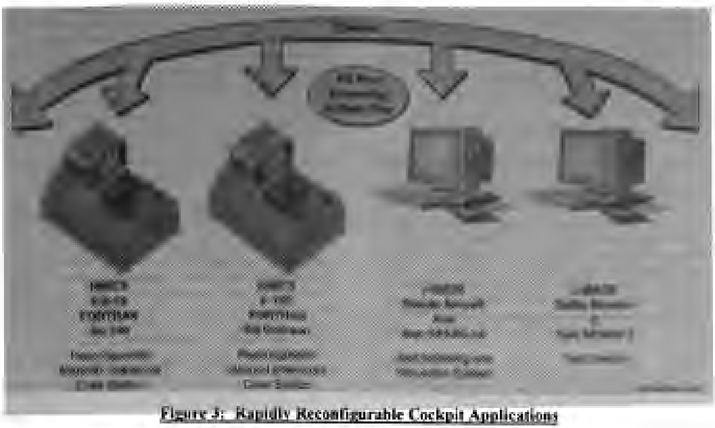
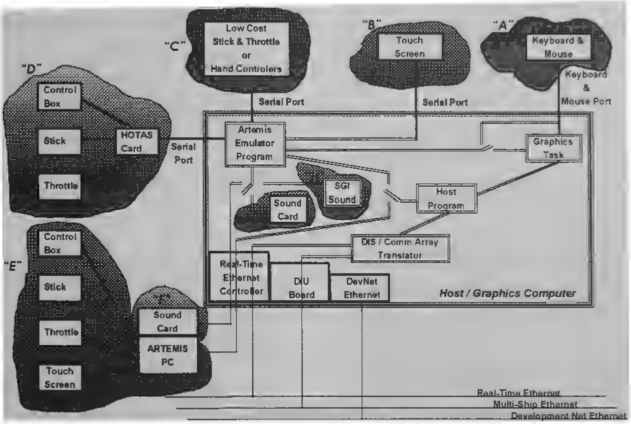


Figure 3: Rapidly Reconfigurable Cockpit Applications



WCB 1 0

Figure 4: Rapidly Reconfigurable Cockpit Configurations

Design

The high level reconfigurable design is illustrated in Figure 4. Hardware Interface methods "A", "B", "C", "D", and "E/F/G/H" are included. Within the "Host/Graphics Computer" system are high level software tasks comprising the Rapidly Reconfigurable Cockpit software configuration. Options "F/G/H" are possible ways of generating simulated aircraft sounds and tones.

Hardware Configuration "A" is a simple keyboard and mouse. Discrete and analog crewstation inputs are mapped simply to provide simulation control inputs in a desktop environment. This design is often sufficient when combined with adequate desktop graphics capability to perform inter-system development and checkout with basic simulation interactivity which is representative of modern aircraft. The graphical representation closely approximates the desired cockpit, and the mouse is used to simulate switch interactions in the cockpit with the graphically represented switches and symbology.

Hardware Configuration "B" provides enhanced realism in interactivity through the addition of a touch sensitive screen instead of, or in addition to, the mouse inputs from Configuration "A".

Touchscreen or mouse inputs are very suitable for reconfigurable simulations which may change very quickly. For example, the Rapidly Reconfigurable Cockpit transition time to switch completely from one aircraft to another, such as from an F/A-18 to a T-45, was designed to be 5 seconds or less. A touch sensitive screen was the only suitable means available to immediately adapt graphical cockpit display interactions to the new or modified cockpit. Also, cockpit layout designers have used the RRC software to move items on the cockpit about, and then fly missions immediately to observe results. Such rapid rearrangement is accomplished handily with the use of a touch sensitive screen.

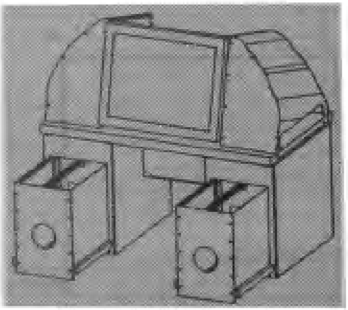


Figure 5: RMICS Construction

Still available in the desktop regime, Hardware Configuration "C" offers a step above the keyboard for basic control inputs. MDA has identified and helped develop several very low cost controller options, similar to game controllers available for modern aircraft game simulations. Modifications were made for added realism in terms of the number of switch inputs on the controllers, as well as the general feel of the controller. The controllers were interfaced through the desktop computer's serial port.

Hardware Configuration "D" introduces flight hardware grips for realistic real-time interactivity. A control box simulating side panel switches is also available. This entry level into the more realistic regime uses a serial interface to the host computer system. This configuration begins to move away from the traditional desktop to more of a table top configuration because of the way the realistic grips can be mounted over the edge of a table.

Hardware configuration "E/F/G/H" consists of Control Input Configuration "E" grouped with one of sound options "F/G/H". Figure 5 below shows the basic RMICS layout, which embodies configurations "E/F/G/H". Configuration "E" is PC based. The control switch box, high fidelity stick and throttle, and touch screen all connect directly to the PC I/O system. Sound Configuration "F" is available when a sound card is placed into the crewstation I/O system. Sound Configuration "G" utilizes a sound card placed into the host computer system, usually a VME backplane in an SGI computer. Sound Configuration "H" directly uses the host/graphics computer's native sound generation capabilities.

The key to seamlessly scaling the fidelity of the interactivity between configurations "A" through "H" is the standardization of the interface. Configuration "E" has the highest fidelity. It resides on a separate PC, and is connected to the host computer by Ethernet. The format of this Ethernet message is used as the standard crewstation interface. The other configurations, "A" through "D", enter the host computer via the serial port, where they are read by a software task which formats the data into a message exactly matching the standard described for Configuration "E". This emulator task uses command line arguments to indicate which means provide which inputs. Through the use of Unified Message Passing (UMP) [5], the host task running on the host computer can not detect any difference between the actual Ethernet message sent in Configuration "E" and the emulated I/O system which sends a message of the same format in Configurations "A" through "D".

Other software tasks running in the Host/Graphics computer system are largely grouped into the Host task and the Graphics task, which may or may not necessarily be collocated on the same actual computer. In the desktop environment, it is desirable to have one computer perform both tasks. In higher fidelity systems, it is usually desirable to tailor the number of computer systems, or processors within a computer system, appropriately to meet the computational and graphics requirements of the particular simulation. Figure 4 shows the preferred digital network connections, using separate networks for Real-Time Ethernet, Multi-Ship (DIS), and development (NFS, FTP, etc.). All communication between tasks uses UMP, which makes the location and distribution of the tasks transparent to each task.

Usage

Various forms of the RMICS have proven useful in many different arenas. Development of new aircraft concepts, as well as modifications to current aircraft, are assisted through RMICS development. Simulations of F/A-18C, F-15E, AV-8B, T-45, T-38, four futuristic cockpits out to the year 2020, a helicopter, a transport aircraft, and others, have all been developed at various levels of fidelity on the RMICS system.

The system is Distributed Interactive Simulation (DIS) Protocol Version 2.04 compliant, and has been demonstrated every year at I/ITSEC since the first RMICS was built in 1993, playing the role of an F-15E and F/A-18C in various scenarios [2]. It has been used to assist in the development and demonstration of the Joint Modeling And Simulation System (J-MASS) program [3]. The mobility of the RMICS system lends itself well to conferences and trade show exhibits, as evidenced by its appearances at conferences such as SO/LIC, NAECON, AFA, AFCEA, SAFE, and I/ITSEC, as well as various demonstrations in the Pentagon, Air Force Bases, and the offices of our virtual aircraft customers.

Conclusion

The design developed during the RMICS program has matured into two basic products: the Reconfigurable Manned Interactive Crew Station (RMICS) hardware system and the Rapidly Reconfigurable Cockpit (RRC) software baseline. The RMICS hardware consists of high fidelity grips, touch sensitive screen, a console based configuration with built-in PC I/O system and standard 27 inch diagonal CRT. An optional additional 27 inch CRT may be mounted above the first CRT for a two channel heads-down and out-the-window presentation. This system has a standardized interface, which allows for easy scalability to lower or higher fidelity interactive environments.

The RRC software provides a wide variety of simulated aircraft, capable of seamlessly scaling from a single CPU system to a multiple CPU system to distributed computer systems using the UMP methodology. Its ability to rapidly switch to various aircraft types makes it an ideal demonstration platform. It can also be tailored to remain as a specific airplane as required.

Future Work

Future work is targeted to improve the lower end desktop interactivity systems, improve the fidelity of the RRC software models, and use JMASS to provide a mainstream simulation software platform for rapidly reconfigurable cockpits. This software platform will be more object oriented by design, bringing the seamless nature of software fidelity scaling up to the same level as the interactivity fidelity scaling capability it currently possesses.

References

[1] Silicon Graphics Incorporated "Periodic Table", 1996.

[2] Beavin, Emrich, Bezdek, "J-MASS DIS Interface Using Ports", NAECON, May 1995.

[3] Emrich, Bezdek, Beavin, "J-MASS Spatial To DIS Port Conversion", NAECON, May 1996.

[4] Burgleman, Maidique, Wheelwright, "Strategic Management of Technology and Innovation", 1996, p 841

[5] Murray, "J-MASS Interconnect Backplane Using UMP", NAECON, May 1994.

Table 1: Interactivity Payoff Table - Estimated Times

Test Example	Test Equipment	Computer System	Digital / Scripted Test Time (Hours)	Interactive Test Time (Hours)	Interactivity Performance Increase
Software Module Functionality	Keyboard/Mouse	SGI Indy	0	0.00	1
Software Switch Functionality	Keyboard/Mouse Desktop RMICS	SGI Indy	0.5	0 08	6
A/A RADAR Display Functionality	Suncom Grip Desktop RMICS	SGI Indy	2	0.25	8
A/A RADAR Dynamic Performance	Flight Hardware Grip Table Top RMICS	SGI Indigo2 Extreme	6	0.25	24
A/A Beyond Visual Range Dynamic System Performance	Console RMICS	SGI Indigo2 Impact	10	0.33	30
A/A Beyond Visual Range Dynamic Engagement	Shell RMICS	SGI Onyx	12	0.33	36
A/A Dynamic Engagement	Partial Dome RMICS	SGI Onyx	40	0.50	80
Visual Range Helmet Designation	Dome RMICS	SGI Onyx	120	1.00	120

Table 2: Interactivity Payoff Table - Estimated Costs

Test Example	Configuration Pricepoint	$65 Per Hour Interactivity Breakeven (Hours)	Depr. Years	Hours P/Week	Digital Cost Per Test	Interactive Cost Per Test	Interactive Savings Per Test	Interactive Tests To Cover Config Pricepoint	Interactive Hours To Cover Config Pricepoint	Interactive Weeks To Cover Config Pricepoint
Software Module Functionality	$ 10,000	0	0	0	$ -	$ -	$ -	0	0	0
Software Switch Functionality	$ 15,000	45	3	40	$ 33	$ 6	$28	542	45	1
A/A RADAR Display Functionality	$ 20,000	43	3	40	$ 133	$ 17	$116	172	43	1
A/A RADAR Dynamic Performance	$ 60,000	39	3	40	$ 400	$ 19	$ 381	157	39	1
A/A Beyond Visual Range Dynamic System Performance	$ 150,000	78	5	40	$ 666	$ 26	$ 640	235	78	2
A/A Beyond Visual Range Dynamic Engagement	$ 200,000	86	5	40	$ 799	$ 28	$ 771	259	86	2
A/A Dynamic Engagement	$ 3,000,000	594	10	40	$2,664	$ 137	$ 2,527	1,187	594	15
Visual Range Helmet Designation	$ 6,500,000	851	10	80	$7,992	$ 351	$ 7,641	851	851	11

AIAA-96-3472-CP

MODELING AND SIMULATION AND THE REVOLUTION IN MILITARY AFFAIRS

Dale C. Hill, Senior Analyst
The Strategic Assessment Center
Science Applications International Corporation
McLean, VA

Abstract

The paper examines the role of modeling and simulation in realizing a Revolution in Military Affairs (RMA). An RMA is defined by the Secretary of Defense's Office of Net Assessment (OSD/NA) as ***"A major change in the nature of warfare brought about by the innovative application of new technologies which, combined with dramatic changes in military doctrine and operational and organizational concepts, fundamentally alters the character and conduct of military operations."*** To understand and exploit these potential changes in the nature of warfare, the Armed Forces and other governmental agencies are conducting seminar wargames, similar to those conducted by the Navy at Newport during the interwar period. To complement these games, there is the need to model and simulate new organizational and operational (O&O) concepts surfaced during the games, especially in light of new and proposed technologies that may be required to support these O&O concepts. Existing models and simulations must be improved and new models and simulation means developed to exploit the emerging RMA within the framework of potential new warfare areas. The challenge is to develop models and simulations to help planners for tomorrow's forces realize and take advantage of the emerging RMA in a cost-effective, realistic manner.

I. Introduction

Military theorists around the world have long noted the historical discontinuities in the conduct of warfare caused by the advent of new technologies and weapon systems. The Soviets called these discontinuities "Military-Technical Revolutions." Recently, analysts in the United States have started calling them "Revolutions in Military Affairs" (RMA). This change in terminology was meant to capture the non-technical dimensions of military organizations and operations, the sum of which provides a large part of overall military capabilities.

The nature of these historical discontinuities is such that warfare after the "revolution" is unlike what went on before in profound and significant ways. Throughout history there have been a number of such revolutions. Gunpowder produced an early military revolution in the Western World, transforming both land and naval warfare. During the mid 19th Century, industrialization revolutionized warfare through the introduction of railroads, the telegraph, the steam engine, rifled guns, and ironclad ships. More recently, the mechanization of warfare during the interwar period led to the development of *Blitzkrieg*, Carrier Warfare, Amphibious Warfare and Strategic Bombing.

In some cases, the changes in technology associated with these revolutions changed not only transportation, communication, and warfare but also entire societies as well. During the Transportation Revolution, for example, railroads altered the economies of nations and also allowed them to move military forces farther and faster and to sustain them longer. Moreover, these societal changes created new vulnerabilities at the operational and strategic levels of warfare which could be targeted by an adversary.

Before proceeding, however, a word of caution is in order. Although we may now stand at the start of a long period in which we may undergo a Revolution in Military Affairs, there are many uncertainties. These uncertainties include: when the transition period might start; how long it might last; what new warfare areas might be developed; and a host of other key questions. In short, there is not an absolute grasp of the scope, pace, and implications of this possible RMA.

Nevertheless, there are useful observations even at this early stage. During and immediately after World War I, forward-thinking military officers such as Colonel J.F.C. Fuller of the British Army and Major Earl Ellis of the U.S. Marine Corps outlined the basic features of Armored Warfare and Amphibious Warfare respectively. They defined these concepts decades before the systems necessary for them to be executed even existed and at a time when the political circumstances of the next war were uncertain. More recently, science fiction writers have been exploring future warfighting capabilities. The novels Starship Troopers by Robert Heinlein (1957) and Ender's Game by Orson Scott Card (1977) alluded to military systems that today equate to artificial intelligence and virtual reality. These future systems appeared to be pure fantasy at the time, but yesterday's fiction has, in many instances, become today's reality. It should be instructive to those engaged in looking to the 21st Century that military officers and authors were able to look into the future and visualize types of warfare that became reality or which are becoming more possible.

II. Lessons Of Past RMAs

Revolutions in Military Affairs have risen from various sources with many, but not all of them, technological. Societal change contributed to a military revolution during the wars of the French Revolution and the Napoleonic era, in which the "*levée en masse*" allowed for the creation of larger, national armies. In the technologically-based military revolutions of the 20th Century, different scientific fields have provided the enabling factor. For example, chemistry and early physics drove many of the advances critical during World War I. In this war of gunpowder, the rate at which weapons fired and the ranges that the projectiles traveled decided the fate of many battles. Advanced physics drove the next RMA, which extended from the mastery of flight to the introduction of radar through the creation of nuclear weapons at the end of World War II. The current RMA may have as its source what has been called new physical principles which focus on technologies such as directed energy. Trends indicate that the next Revolution in Military Affairs may have a biological source. Some manifestations of these biological advances may include nanotechnologies, neural networks, distributed systems, bio-sensors, bio-electronics, and performance-enhancing drugs.

New technologies and systems significantly influence RMAs, although the resulting revolution could take any of a number of forms. Innovations during the interwar period – Armored Warfare by the German Army, Amphibious Warfare by the U.S. Marine Corps, Carrier Warfare by the U.S. Navy, and Strategic Bombing by the U.S. Army Air Forces – have been characterized as "combined system RMAs." Their revolutionary nature derived from a collection of military systems put together in new ways to achieve a revolutionary effect.

A different type of RMA, the "single-system RMA" can best be exemplified in the 1940's and 1950's, in which a single capability – unleashing nuclear power – drove the revolution. Another example of a single-system RMA is the gunpowder revolution, in which gunpowder transformed land and naval warfare through the use of siege guns, field artillery, infantry firearms, and naval artillery.

Evidence suggests that the revolution unfolding today is neither a "combined system" nor a "single-system" RMA but rather an "integrated system" RMA. The outlook is for the rapid evolution of new technologies eventually leading to the development of several advanced military systems. These systems, when joined with their accompanying operational and organizational concepts, will become "integrated systems." In contrast to developments during the interwar period, this system of systems approach will aim to take advantage of the cumulative effect of employing each of the new capabilities at the same time. In World War II, each new form of warfare primarily took place in its own operating medium – Armored Warfare on land battlefields, Strategic Bombing in the air over homelands, Carrier Warfare at sea, and Amphibious Warfare across the intersection of land and sea. Only occasionally did one warfare area interact with or support the others. In the current RMA, the integrated employment of all the new systems will be essential to take advantage of their true utility.

Nonetheless, this forecast does not exclude the possibility of a "single system" RMA. To avoid strategic surprise, we must continue to think about breakthroughs in critical areas such as information technology, biogenetics, and others. Unforeseen advances in these areas could bring about a sudden, significant, and solely-owned military advantage to the country that achieves a breakthrough. The same holds true for a "combined system" RMA. Furthermore, there is also the possibility that the Information Revolution may result in far-reaching societal changes putting us on the path of a "Social-Military Revolution." Such a revolution implies profound, but somewhat different, implications for the changing nature of warfare.

It is important to remember that technologies and systems enable but do not cause military revolutions. What often "unlocks" an RMA is the new and innovative way in which existing technologies are integrated into the military through organizational and operational (O&O) changes. Past military revolutions have been driven by ***requirements*** that have motivated military organizations to innovate in order to overcome the limitations of existing practice. Such strategic, operational, and tactical requirements determined whether technologies were adopted and how they were employed. Without them, stagnation can prevail even in states possessing technologies with revolutionary implications.

An example of this principle is the gunpowder revolution of the 16th and 17th Centuries. In Europe, gunpowder weapons fundamentally changed the conduct of all existing areas of warfare (maneuver, siege, and naval) on account of the constant competition between rival states of roughly equal military power. Imperial China developed gunpowder and firearms a century before Europe possessed them, but then stagnated in the gunpowder revolution largely because of its vast population; it overcame any land or sea threats through sheer weight of numbers. In contrast, the smaller states of Asia made significant innovations in response to pressing military requirements. In 16th Century Japan, where rival warlords strove for dominance, the rise of firearm-equipped infantry and the use of volley fire mirrored developments in Europe. Korea in the 1590's responded to the threat from reunified Japan by developing its "turtle ships" -- ironclad, cannon-armed galleys that provided a technological advantage that proved essential to defeating three successive Japanese invasions.

Another example is the interwar period, in which all of the major powers possessed the same technologies but only a few countries created new operations concepts and organizations to exploit them. For example, the development of Carrier Warfare took place in the United States and Japan for the purpose of fighting major naval engagements in the Pacific. On the other hand, Carrier Warfare withered in Britain's Royal Navy, as its main tasks were fighting in confined waters such as the Mediterranean Sea and in combating German commerce raiders on the high seas. In the case of Armored Warfare, only the Germans employed tanks, radio, and airplanes in new ways in 1940 even though all of the essential technologies had been available since World War I. This situation was due in part to Germany's unique strategic problem of being surrounded by enemies leading to an operational requirement for rapid offensives to defeat enemy states quickly, with the emphasis on offense dictating tactical requirements for mobility, firepower, and protection. Germany thus integrated tanks, infantry, and artillery in its *Panzer* divisions and supplemented them with close air support. France and Britain, constrained by defensive strategic and operational concepts on land, did not innovate and subsequently paid the price in early 1940; although each possessed weapons that matched or even exceeded, in both quantity and quality, what the Germans fielded.

Past RMAs hold another significant lesson for the current RMA. In past RMAs many of the key systems were already used in combat or in civilian applications decades before significant changes occurred in military organizations. For example, railroads began carrying commerce in the 1830's, but in the 1860's only the Prussian army under von Moltke used them to facilitate intricate mobilization plans that conferred a significant operational advantage at the start of a campaign. Similarly, tanks, radios, and close support aircraft were used in quantity in World War I, but their true potential was not realized until the Germans devised new organizational and operational concepts for them in the 1930's. Likewise, the implementation of revolutionary O&O concepts in this RMA may require a long time even though most of the key systems probably are already in development or have even been used in combat. We need to start thinking immediately about the shape of future warfare in order to capitalize on the RMA in a timely fashion.

III. New Warfare Areas

The current RMA, like the interwar period, will probably involve the emergence of multiple new warfare areas; defined as a form of warfare with unique military objectives and characterized by association with particular forces or systems. As alluded to earlier, warfare areas that emerged in the interwar period and which played a very big role in World War II are: Armored Warfare; Carrier Warfare; Amphibious Warfare; and Strategic Bombing.

There are currently four potential new warfare areas identified by OSD/NA from which the current RMA may well emerge. They are: long-range Precision Strike; Information Warfare; Dominating Maneuver; and Space Warfare. Other areas, yet to be identified, could also emerge by the year 2020, the "target" study date chosen because it helps those involved to think "outside the box" and also pushes planners well beyond the budget cycles which often drive conceptual thinking.

The warfare areas identified are likely to emerge in the long run but will not necessarily be developed fully in the near future. Doctrinal development is a long and uncertain process, and military history offers numerous examples of unexploited warfare areas – concepts intended to revolutionize warfare that did not come to fruition. Technological limitations, conflicts with prevailing doctrine, or lack of strategic purpose derailed these developments.

In the late 19th Century, the *Jeune École* in France sought to exploit an emerging weapon, the torpedo, to contest British sea control with small, cheap torpedo boats for commerce raiding and coastal defense. This attempt to create a new warfare area led to a decade of doctrinal uncertainty in naval warfare but failed by the turn of the century owing to the ineffectiveness of the primitive torpedoes and torpedo boats, the rising influence of Mahan, and the emergence of the Anglo-French Entente.

After the Korean War, General James Gavin and others in the U.S. Army sought to create a new form of land warfare using the helicopter. Seeking greater strategic mobility between theaters and a "mobility differential" over the battlefield, they envisioned helicopter-equipped units that could rapidly deploy in a crisis and use their superior mobility in the cavalry roles of scouting, pursuit, and delaying actions. They succeeded in making helicopter aviation a significant part of the Army, but the helicopter evolved as part of a combined-arms team rather than as the basis for autonomous units, thus the Army fielded only one Air Assault division during and since the Vietnam War. The vulnerability of helicopters to air defenses and the predominance of armor and infantry in existing doctrine each contributed to this result.

Nevertheless, short-term technological and doctrinal barriers will not diminish the ultimate importance of a new warfare area. There are past examples of warfare area concepts that were abortive in one context but resurfaced in other settings with the emergence of the right enabling technologies or doctrinal pressures. The ideas of the *Jeune École* appeared again in Germany during World War I, when practical submarines were the enabling technology and the need to strangle British commerce provided the doctrinal pressure. Similarly, the nascent armored warfare concepts of J.F.C. Fuller and the Salisbury maneuvers went undeveloped in Britain but re-emerged in the German Army. The fact that concepts discarded by Britain and France provided the basis for U-Boat warfare and the *Panzer* divisions illustrates that fundamentally sound concepts will eventually be exploited – if not by those who first envisioned them, then possibly by a future adversary.

Of the four potential new warfare areas, Precision Strike is the most developed conceptually, although even here much analytic work remains to be done. Much work has also been done in the area of Information Warfare, yet it remains a poorly understood concept. Analysis of Dominating Maneuver and Space Warfare has just begun.

Precision Strike

Precision Strike may well be the most thoroughly understood new warfare area of the emerging Revolution in Military Affairs. This is true because the United States has been a leader in the development and deployment of such systems since the 1970s. The Persian Gulf War demonstrated the potential for deep strike systems to not only create a maneuver differential, but at least potentially to be decisive in themselves. Precision Strike, in the context of the unfolding RMA, is "the ability to locate high value, time sensitive, fixed and mobile targets and to destroy them with a high degree of confidence; and, to accomplish this within operationally and strategically significant timelines while minimizing collateral damage, friendly fire casualties, and enemy counter-strikes."

The potential effect of Precision Strike can be seen in the recent dramatic increase of capabilities to strike strategic targets. Colonel John Warden, the Air Force planner credited as being the architect of the Desert Storm air campaign, cited the following comparison: in 1943, the U.S. 8th Air Force prosecuted only 50 strategic targets during the course of the entire year* while, in the first 24 hours of Desert Storm, the coalition air forces prosecuted over 150 strategic targets† representing greater than a 1000 fold increase over 1943 capabilities. Recently, General Fogleman, the Air Force Chief of Staff, made the following observation: U.S. forces in the near future "may be able to engage 1,500 targets in the first hour, if not the first minutes, of a conflict.‡" If made in the first hour, this would represent a 2,500-fold increase over Desert Storm capabilities.

It is envisioned that Precision Strike will be able to achieve effects similar to those of nuclear weapons but without the attendant risk of escalation to intolerable levels of destruction. When directed against targets comprising enemy centers of gravity, Precision Strike might itself prove decisive. For it to do so, however, requires a much clearer understanding than we have had in past wars about what constitutes enemy centers of gravity and what it takes to affect them.

The essence of Precision Strike is the ability to sense the enemy at operational and strategic depth and recognize his operational concept and strategic plan; to select and prioritize attacks on enemy targets of value. All of this is intended to achieve decisive impact on the outcome of the campaign. In 2020, Precision Strike technologies may well create the potential to achieve strategic effects at intercontinental distances much as nuclear tipped ICBMs did during the Cold War.

At the same time, there needs to be an equal effort in developing new operational concepts and organizations for the application of Precision Strike. As in other new warfare areas, it may be that the greatest payoff will come from new O&O concepts, not from systems.

Information Warfare

Another emerging warfare area is that associated with information systems, their associated capabilities, and their effects on military organizations and operations. This is termed Information Warfare (IW) and is defined as "the struggle between two or more opponents for control of the information battlespace." The objective of IW is to deny the enemy critical knowledge while helping to secure friendly information flow with the goal being dominant battlespace awareness.

At the national level, Information Warfare could be viewed as a new form of strategic warfare, where one of the key issues is the vulnerability of socio-economic systems and a key consideration is how to attack the enemy's system while protecting one's own system. At the military operational level, IW may contribute to major changes in the conduct of warfare, and therefore, one of the key issues is the vulnerability of military command, control, communications, computer, intelligence, surveillance, and reconnaissance (C^4ISR) systems. Again, the key consideration is how to attack the enemy's system while protecting your own.

As information capabilities are increasingly assimilated into a military structure with the focus on establishing and maintaining an "information advantage" as a war-winning strategy, the vulnerabilities of those forces, and, ultimately of the nation itself are changed. A force structure that will implement "Information Warfare" 25 years from now may well be different from today's military in more ways than just its equipment. Moreover, the character of warfare may change in ways that affect intelligence and crisis and wartime decision-making.

Some of the changes might include the whole issue of deciding that a war has begun. It is not clear at this time whether IW measures taken by a potential adversary at the outset of a war would be readily detectable. The question of "how do you know you are at war" may be difficult to resolve in view of the potential ambiguity associated with Information Warfare. Deciphering barely perceived actions along with the plausible deniability of occurrences are not new phenomena, hence the term "fog of war" still applies. But, the rapid growth of interconnections manifested already in communications, banking, and other areas present opportunities to do grievous harm, quickly and with

* Represents "strategic" targets within the pre-1939 borders of Germany struck by heavy bombers.[1]

† There were 264 "strategic" strikes (not the same as targets) on Day 1 of Desert Storm.[2]

‡ Address to the Air Force Association's Air Warfare Symposium, February, 1995.

no warning, and with a minimal "signature." Accordingly, the analysis of indications and warning that mark the outset of warfare must change.

In addition to its inherent ambiguity, Information Warfare in 2020 also portends a very different set of potential responses by the United States to an adversary detected acting in a hostile or potentially hostile fashion. The IW measures that the U.S. might take could require quite different policies, with regard to Rules Of Engagement, than have previously been contemplated. This involves both good news and bad news. The good news is that there may be new tools, short of lethal attacks, available to signal an adversary that warfare with the U.S. would be a bad idea. The bad news is, with the inherent ambiguity of IW, it might be difficult to ensure that the enemy received the desired intentions.

Although countering an adversary's command and control has always been a feature of warfare, the continental United States has been somewhat invulnerable to such measures. One clear implication of warfare in 2020 is that almost any enemy will try to degrade U.S. information systems. Paradoxically, although the technology of information systems is becoming more capable and sophisticated, it is, at the same time, becoming harder to secure the U.S. information infrastructure from attacks. One potential vulnerability is the fact that Information Warfare generates problems at the national level rather than just for the Department of Defense. The problem goes beyond the armed forces to the entire national security infrastructure. As the international information infrastructure grows and elaborates, its reach expands beyond the control of any single entity or any single nation. Thus, the infrastructure is beyond the control of those who use it, and has access points at a myriad of places for others to enter the system.

In dealing with the information "revolution" which is affecting the military today, the U.S. military services seem to be engaged in improving their current communication channels. That is, they are striving to improve performance elements ***within*** the current organizational structure. They have yet to address the implications of systems and capabilities that do not fit within the current structure. This is a fundamental issue. The military traditionally has viewed information services, including intelligence and communications, as ***supporting*** inputs to the actual warfare functions of fires, maneuver, strike, and the like. However, information warfare might not always be a supporting function, it might take a leading role in future campaigns. By 2020, at least in some militaries, the requirements of the battlefield will be such that traditional hierarchical command and control arrangements will be obsolete. In most organizations today the trend toward decentralization is already well-established. Information technology is making distributed systems commonplace and "virtual organizations" are growing like cultures on a petri dish. The rapid rate of growth of these types of new organizational entities would seem to suggest strengths which the military would be wise to examine.

Dominating Maneuver

One of the more recently identified potential new warfare areas is Dominating Maneuver. Maneuver has always been an essential element in warfare, but the RMA potentially offers the ability to conduct maneuver on a global scale, in a much-compressed time frame, and with greatly reduced forces. Dominating Maneuver is defined as "the positioning of forces, integrated with Precision Strike and Information War operations, to attack decisive points, defeat the enemy center of gravity, and accomplish campaign or war objectives." While Precision Strike and Information Warfare are destroying enemy assets and disrupting his situational awareness, Dominating Maneuver will strike at the enemy center of gravity to put him in an untenable position, leaving him with no choice but to accept defeat or accede to the demands placed on him.

War is typically non-linear, meaning that the smallest effects can have unpredictable, disproportionate consequences. In meteorology, non-linearity is illustrated through the "butterfly effect" -- a butterfly flapping its wings in the southern hemisphere can set off a string of reactions that eventually result in a violent storm in the northern hemisphere. In the early 19th Century, Clausewitz made similar observations when discussing the formulation of successful strategy. He wrote that victory comes not through winning battles or inflicting attrition but through attacking the enemy center of gravity. Depending on the situation, the center of gravity could be the enemy's army, his capital, his leaders, or his principal ally.

In the course of the 20th Century, it appears that the complexity of warfare has increased as military forces and their logistical and political underpinnings have become more complicated. With increasing complexity, the non-linear nature of war is likely to increase. Dominating Maneuver seeks to exploit the increasing complexity and non-linearity in warfare by striking directly at the enemy center of gravity in order to disrupt his cohesion and cause his swift collapse.

Dominating Maneuver is distinct from maneuver in several ways. Joint Publication 1-02 "Department of Defense Dictionary of Military and Associated Terms" defines maneuver as the "employment of forces on the battlefield through movement in combination with fires, to achieve a position of advantage with respect to the enemy in order to accomplish the mission." Dominating Maneuver refers to the ***positioning*** of forces, not necessarily their employment; they can be positioned anywhere in a theater, not necessarily on the battlefield. It goes beyond "combination with fires" by integrating its effects with the effects from Precision Strike and Information Warfare. Its ultimate purpose is directly to achieve ***campaign*** and ***war*** objectives, transcending the role of ordinary maneuver.

Dominating Maneuver does not require superiority at all points in the battlespace or imply domination of the entire maneuver. Competitors in 2020 could well challenge U.S. national interests in regions where they enjoy the advantage of close proximity, with the U.S. denied a lengthy build-up period to marshal forces or access to a continental infrastructure to support forces in the theater. Under these circumstances, the continental United States (CONUS) may be the principal base of operations, making the maneuver battlespace orders of magnitude larger than it was in Desert Storm. Dominating Maneuver could allow ground forces to operate successfully in situations where they cannot dominate the entire battlespace.

Dominating Maneuver will require new operational concepts that take into account the decisive importance of time, making future maneuver more simultaneous than sequential. Attaining operational and strategic objectives through simultaneous Information Warfare, Precision Strike, and maneuvers against the enemy's critical points rather than through a series of pitched battles against enemy forces will be essential.

The German invasion of Norway in April 1940 was a campaign waged successfully in a way analogous to Dominating Maneuver, although on a smaller scale. A single airborne maneuver into Oslo on the 9th of April induced the surrender of the city's garrison by creating the perception that their cause was hopeless. Fighting continued on the fringes of Norway for six weeks, but the airborne maneuver led directly to decisive results. The maneuver gave the German commander a time advantage during which he could reinforce faster than the Norwegians could mobilize or the Allies could deploy. Moreover, the surrender of the Oslo garrison precipitated the capitulation of the Norwegian monarchy and forced the Allied decision not to become heavily engaged on the Scandinavian peninsula.

The Inchon landing during the Korean War was another operation that illustrates the principles underlying Dominating Maneuver. General MacArthur's plan to capture Seoul through an amphibious landing at Inchon struck at one of the critical vulnerabilities of the North Korean forces – their dependence on the transportation bottleneck at Seoul. Instead of gradually rolling the North Koreans back from Pusan, MacArthur planned to cut them off and put them into an extremely vulnerable position. The landing paralyzed the already overstretched North Korean forces, and they broke into disorganized fragments that retreated in disarray, incapable of mounting serious resistance.

The organization and tactics of ground forces capable of Dominating Maneuver are difficult to visualize today. Some have suggested that 21st century variants of the so-called "*Hutier*" tactic developed by the Germans in World War I -- "Stingray" or "infestation" tactics -- would be useful. Such tactics would combine deception and bombardment with infiltration and attacks against strong points. The ground forces may be a small number of infantry or special operations forces, delivered deep in enemy territory by air and equipped with high-technology linkages to space-based or atmospheric strike systems; in effect, acting as part of a sensor-shooter network.

Progress in these conceptual and technological areas will enable maneuver to play a significant role in the RMA. It is possible that Precision Strike and Information Warfare will make maneuver unnecessary in certain situations, or that enemy progress in these areas will make maneuver difficult. Nevertheless, maneuver will be essential against an enemy unwilling to concede defeat unless the U.S. defeats centers of gravity that cannot be attacked without maneuver forces. Dominating Maneuver may provide the *coup de grâce* in future wars, and in other situations may serve as the enabler for war-winning Information War or Precision Strike operations.

Space Warfare

Space Warfare is the fourth future warfare area and is currently defined as "the exploitation of the space environment to conduct full-spectrum, near-real-time, global military operations." It includes facets of the other three warfare areas but has the potential to become a qualitatively distinct warfare area in its own right. The U.S. military's increasing reliance on support from space-based systems for its everyday operations and especially during times of conflict has highlighted the importance of space operations. However, space assets could provide more than support for the terrestrial warfighter in the future. The space environment offers the possibility of conducting worldwide military operations in a greatly reduced timeframe.

The evolution of space operations is comparable to the development of air warfare, which similarly exploited inherent advantages in altitude and speed. Aircraft filled an essential role in supporting the ground and naval forces in the First World War through observation, anti-observation, ground attack, and communications. Between the wars, larger aircraft came into service in the form of civil and military transport, a capability that was greatly expanded during World War II. Moreover, during the interwar period, the U.S. and Great Britain developed airpower as a means of leapfrogging conventional ground and naval battles to enable direct strikes on the enemy's ability to wage war. Although the theory of the effects of strategic bombing met with limited success in World War II, the concept subsequently culminated in the development of the intercontinental nuclear deterrent of the Cold War.

Like the air operations of World War I, space operations currently provide support essential for the successful operations of terrestrial forces. Satellites enable near-real-time capabilities in worldwide communications, sensing, and navigation. These capabilities, analogous to the roles of the observation balloons and aircraft in WW I, may make possible dominant battlespace awareness and coordination of a global Precision Strike architecture. An effective anti-satellite (ASAT) capability could lead to achievement of aerospace control in order to deny an opponent the ability to operate in or from space. An ASAT system would follow in the footsteps of the first pursuit aircraft that dueled for control of the air over the trenches of the First World War and is a logical extension of the current role of air superiority fighters and developing theater air defense systems.

However, space operations will also greatly differ from air operations. First, the "geography" of space is fundamentally different from that of the Earth's atmosphere. Orbital mechanics require operating speeds (17,000 miles per hour) that far surpass those currently achievable in the atmosphere. Thus, properly placed and employed space assets could perform missions in much less time than state-of-the-art aircraft. One possible mission would be to use space forces to project power to directly achieve national objectives (operational or strategic) in a particular theater. Space-based Earth strike systems employing high-density "rod" and based on satellites or deployed from transatmospheric vehicles could enable precision strikes whose quantitative advantage in speed would result in a qualitative difference in capability. Although currently limited, future capabilities in space transport may also make possible the movement of critical forces and equipment from CONUS to a theater in timeframes an order of magnitude faster than what is possible with current sea and air transport. Thus, space operations may provide important advantages in time-critical situations. Further, the altitude advantages provided by space greatly improve surveillance and reconnaissance coverage of the Earth and, as a result, could offer the means to command and control operations in theaters where distance and terrain complicate or confound terrestrially-based systems.

Space warfare will likely become a distinct warfare area in its own right when there is need to conduct military operations in space to obtain solely space-related goals (not missions that are conducted to support Earth-based operations). For example, if the United States becomes dependent on resources unique to space (such as Helium-3 on the Moon), it may be forced to develop technologies and operational concepts to support and defend space-based industries, command and control nodes, or colonies that are independent of Earth. In such situations, space operations would be altogether removed from any congruence with traditional air operations and would undoubtedly become a distinct warfare area.

The Emerging RMA

The above description of the Revolution in Military Affairs is neither definitive nor conclusive. The discussion is intended primarily to stimulate thinking; thinking in unique and more meaningful ways about how warfare in the 21st Century may be fundamentally different than it is today; and, of equal importance, evaluating what should be done now to prepare for that eventuality.

The emerging RMA will likely be the result of the intersection of the four new warfare areas (see Figure 1) discussed above, although it will probably not be equal portions of each. Precision Strike will blind, immobilize, and hold the enemy at a distance while targeting and destroying critical, time-urgent targets. Information Warfare will deny an enemy critical knowledge of both his own forces as well as those which oppose him thus enabling friendly dominant battlespace awareness. Dominating Maneuver will position the right forces in the right place and at the right time to force the enemy's operational and strategic collapse. Finally, Space Warfare will allow conduct of full-spectrum, near-real-time global military operations from space and eventually evolve to include achieving objectives in the space environment.

Warfare Areas in the Emerging RMA

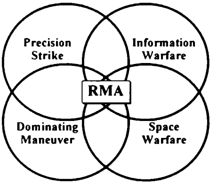


Figure 1

IV. Preparing for an RMA

To achieve a Revolution in Military Affairs, the U.S. military leadership must foster innovative thinking, encourage changes throughout the organization, and develop a strategy for exploiting and pursuing any RMA. Currently, the Services and other DoD organizations are using seminar wargames to place game players in a non-threatening environment with a conceptual force structure and with the challenge of accomplishing a military task in new and innovative ways. These games, like the 300+ exercises conducted at Newport during the interwar period, are helping to identify new and innovative organizational and operational concepts that could provide the key to unlocking the potential in an RMA. However, as these concepts are revolutionary in nature and may therefore require major changes in force structure and, potentially, in the very roles and missions of the Services themselves, they must be examined in greater detail prior to being incorporated into current military force structure and doctrine.

V. Modeling and Simulation and The Emerging RMA

Hence the role for modeling and simulation (M&S) in helping to explore the RMA. In the investigation of the RMA, M&S may help quantify the potential step-function increase in military capabilities possible through new and innovative O&O concepts employing new technologies and weapon systems.

However, the M&S community is not currently equipped to assist in the accurate assessment of warfare in the current military environment, let alone any potential future RMA.

Shortcomings & General Challenges

Critical shortcomings in M&S include the effects of C^{4}ISR and their role in planning and conducting offensive and defensive operations as well as the results of the degradation of C^{4}ISR capabilities on the conduct of military operations. There is also difficulty in assessing the effects of time and what an increasing operational tempo or the simultaneity of effects mean to an opposing force. Another shortcoming is how to model the maneuvering of forces to gain a disproportionate effect and how this can disrupt an adversary (*e.g.*, we still can't model the *Blitzkrieg*). Further, there is little understanding of the operational effects of using weapons of mass destruction on forces (*i.e.*, operating for extended periods in protective gear), on equipment (*e.g.*, EMP effects), and population (especially bio-weapons). Finally, we cannot model the value of morale, leadership, training, or the "fog of war."

The shortcomings result from two factors: the complexity of today's operations are increasing faster than the M&S community can quantify; and models are not currently equipped to accommodate the increasing levels of precision central to future conflicts on a theater scale.

The complexity issue arises from the battlespace integration of all aspects of military operations to achieve theater objectives; wars no longer break down into neat little pockets of conflict with results that can be aggregated to the next higher level until the final outcome can be determined. The results of operations conducted in the air, land, maritime, and space environments by all forces involved are highly interdependent with the outcome in one environment impinging on the outcomes in others. Thus, the modern analogy to the adage "For want of a nail, the shoe was lost..." might be "For want of a GPS satellite, the cruise missile was lost..."

The precision issue stems from the difficulty of reconciling the trade-off between large degrees of uncertainty within a given area (*i.e.*, 10 km^2 hex-shaped regions) and overloading the system with so much detail that it makes it all but impossible to generate useful results in a timely manner. At the same time, precision can be a detriment because there is no accounting for actions taken by individuals or units which can have operational or even strategic effect. An historical example is the commander in World War II who, without notifying higher headquarters, turned from his assigned objective when he realized his unit could seize the bridge across the Rhine River at Remagen. This action provided the Allied ground forces their first crossing point of this strategic barrier and ultimately provided a permanent foothold in Germany. Without human intervention in a computer model, this same unit would have kept advancing toward its assigned objective without regard for the opportunity at hand.

As mentioned earlier, key in the RMA effort are innovate organizational and operational (O&O) concepts which are often employed with existing technologies. Unfortunately, O&O concepts do not lend themselves to quantification as easily as tangible data points like probability of kill or rate of movement. Analysis of O&O concepts introduces the dilemma of distinguishing exactly what is responsible for the outcome of the simulation. That is, did the outcome hinge on a "Silver Bullet" weapon system, an inherent imbalance of the opposing forces, a mistaken assumption, *etc*?

Figure 1 depicts the RMA as emerging from the intersection of four new warfare areas and assumes a step-function increase in the effectiveness of military operations. Such an increase may prove very difficult to quantify as it involves a large number of variables within a conflict and regime that is not fully understood. For instance, how does one measure the effectiveness of an Information Warfare attack on a given system (*e.g.*, activating a Trojan Horse virus that inserts false data in the system's output), let alone the effect that same attack might have on a "system of systems" of which the attacked system is a part?

Some of the new warfare areas (*i.e.*, space and IW) will probably involve new weapon systems and munitions (*e.g.*, EMP generators, Trojan Horse viruses, directed energy) operating at a tempo that is several orders of magnitude greater than what is possible today. However, these weapons and their measures of effectiveness have yet to be developed. At the same time, the increased operational tempo possible in an RMA regime could create unknown effects on the dynamics of warfare. Further, RMA weapons and the necessary C^{4}ISR architecture supporting them will make an already complex inter-relationship even more difficult to understand and quantify.

Finally, simulations must be expanded at both the training as well as the analytic levels. The expansion at the training level should be undertaken to allow warfighting units to participate in theater air, land, naval, or space campaigns built from the individual "battles" currently being simulated in each of the aforementioned environments. The expansion in the analytic arena would be to bring together the various campaigns being simulated (air, land, naval, and space) to create a theater level simulation that encompasses all of these environments.

Specific Challenges of an RMA

Within the new warfare areas of an RMA are challenges to the M&S community. Below are just some that are apparent today.

Precision Strike

This new warfare area is most amenable to realistic simulation with attrition oriented measures of effectiveness (MOEs) (*e.g.*, kills per sortie, loss-exchange ratios). However, the potential RMA step-function increase in effectiveness cannot be adequately captured because: the range, number, and speed of precision strikes call into question traditional MOEs; the paralysis of an adversary (a psychological result) cannot be assessed in a typical model; and strikes against critical nodes (*e.g.*, leadership and C^4ISR architectures) are difficult to measure as is the time element that allows simultaneous strikes *vs* precisely sequential strikes.

Information Warfare

IW is perhaps most critical across all the other warfare areas but is the least understood in terms of quantifiable analysis. This is because there is no clear understanding of RMA C^4ISR architectures, their capabilities, vulnerabilities and the effects of different levels of degradation. There may also be cascading effects where second-order effects take on greater significance because of the tremendous integration of future C^4ISR systems. Further, time delays, which are harder to identify and judge, may have unexpected and important implications with their overall impact greater than that realized from outright destruction. Finally, IW includes spoofing, jamming, disinformation, information capturing, and "soft kills." All of these different means of "attack" are difficult to model.

Dominating Maneuver

Like Precision Strike and IW, a Dominating Maneuver may seek to affect the psyche of an adversary by seizing a center of gravity which will lead to collapse of the enemy's will to fight. Such an outcome is not currently quantifiable. Today's models have difficulty assessing the effects of forces maneuvering in unexpected locations so the prospect of seamlessly moving forces from out of theater to the battlefield, and prepared to fight in such a manner so as to affect an enemy's performance, presents an extraordinary assessment challenge. Additionally, Dominating Maneuver is reliant upon the success of other military operations, making the ultimate importance of the maneuver even harder to gauge.

Space Warfare

The overarching presence of space capabilities rapidly elevates Space Warfare from the operational to the strategic level of war. This further complicates M&S and establishing MOEs which focus on the tactical and operational levels of warfare. Also, since space is currently militarized, and not weaponized, MOEs of future space capabilities are not well developed. Although M&S of current space operations (launch, recovery, and orbital operations) of space payloads is well developed, evaluating new roles for space (*i.e.*, beyond providing secure and assured navigation, communications, and sensing capability) to a terrestrial warfighter, is not. Compounding this is the fact that experts differ and studies have fallen short on assessing the long-term effects of space hazards, making them difficult to quantify. For instance, there is a range of effects (varying from what could be termed a minor headache to a major migraine) believed to occur from an electromagnetic pulse in space resulting from the high-altitude burst of a nuclear device. Additionally, modeling of space debris has primarily been on the effects of limited, small-scale conflicts with a few (<dozen) engagements *vs* multiple, large-scale conflicts with many (>hundreds) engagements.

VI. Conclusion

Existing models and simulations must be improved and new ones developed to help explore and exploit the emerging RMA. Improvements must incorporate new systems and also examine and test how these systems may be employed in new and innovative organizations and in new and innovative ways within the framework of potential new warfare areas. The challenge is before developers of future modeling and simulations to help the military planners for tomorrow's forces realize and take advantage of the emerging RMA.

[1] Extracted from Kit C. Carter and Robert Mueller, The Army Air Forces in World War II: Combat Chronolgy, 1941-1945 (Washington, DC; Office of Air Force History, 1973): 76-242

[2] Extracted from Elliot A Cohen, *et al*, Gulf War Air Power Survey (Washington, DC; GPO, 1993): *Statistical Compendium*, Table 186, "Daily Strikes by Master Target List"

MODELING AND SIMULATION AND C^{4}I: THE QUEST FOR DOMINANT BATTLESPACE AWARENESS

James A. Hazlett
Strategic Assessment Center
Science Applications International Corporation
McLean, Virginia

Abstract

Paper discusses the potential use of modeling and simulation (M&S) in realizing Dominant Battlespace Awareness (DBA). Commanders and scientists have been hypothesizing as to what might be possible in the conduct of warfare on a battlefield where one might conceivably have "total awareness" or a "complete" picture of everything of consequence in the battlespace. Dominant battlefield awareness will make non-linear warfare possible. Battle fronts will lose their significance and powerful, threatening targets may become irrelevant, or merely inconsequential. Information and decision-making processes and apparati may become the real targets of significance, as forces practice "derivatives warfare"—leveraging precision actions, both through physical destruction and information warfare, for limited targets and limited gains, that, in combination, totally overwhelm the adversary's ability to fight effectively. Modeling and Simulation may be the key to converting C^{4}I into Dominant Battlespace Awareness. This use of Modeling and Simulation will affect strike and mission planning, defensive operations, targeting, logistics, operations and command and control. It could conceivably lead to a new type of warfare for the twenty-first century and a true Revolution in Military Affairs (RMA).

I. Introduction

The former Vice Chairman of the Joint Chiefs of Staff, Admiral Bill Owens, and others have been hypothesizing recently as to what might be possible in the conduct of warfare on a battlefield where one might conceivably have "total awareness" or a "complete" picture of everything of consequence in the battlespace. The realization that this may not yet be fully possible, and that potential adversaries could also possibly achieve a similar awareness has led to the creation of the term "dominant battlespace awareness (DBA)." DBA suggests that although one's awareness of the battlespace may not be fully complete, it is better than the adversary's and therefore contributes to success over them.

What has not received a lot of discussion, yet, is how one might best model this battlespace awareness and how it might most usefully be communicated to the commander, the operator, and the warrior. What we call C^{4}I today—command, control, communications, computers and intelligence—provide, and will continue to provide, the information necessary to achieve this awareness. Modeling and Simulation may be one of the keys to converting C^{4}I into true Dominant Battlespace Awareness. Fused intelligence, possibly linked with prepackaged digitized terrain and synthetic symbology, may finally give commanders and warriors a more complete picture of the battlespace, that far exceeds that possible from any single source. Modeling and Simulation may also be a missing link in successfully automating decisionmaking processes. Modeling and Simulation can be used to greatly enhance what we do know, but cannot perceive, about targets with uncorrelated information; and help to clearly to identify the "real unknowns"—those threats about which we know nothing.

This use of Modeling and Simulation will affect strike and mission planning, defensive operations, targeting, logistics, operations and command and control. It could conceivably lead to new types of warfare, from Military Operations Other Than War (MOOTW) through global warfare, for the twenty-first century and a true Revolution in Military Affairs (RMA).

II. Dominant Battlespace Awareness (DBA)

Dominant Battlespace Awareness can be defined as conceivably having "total awareness" or a "complete" picture of everything of consequence in the battlespace, regardless of the nature and level of conflict. DBA is usually discussed with regard to a particular area of battlespace. This battlespace is usually portrayed as a cube, 200 nautical miles on a side, with an unspecified height and depth. In the May 1995 issue of the Naval Institute PROCEEDINGS, Admiral Owens describes battlespace awareness as follows:

> *Battlespace Awareness* rests on the sensing and reporting technologies and includes both the platforms and sensors we associate with intelligence gathering, surveillance, and reconnaissance—and reporting systems that provide better awareness of our own forces, from in-transit visibility and logistics flows to the location, activity and status of our units, allied units, and noncombatants. Included are an awareness of the weather, terrain, and electromagnetic characteristics of any arena in which we may use forces.*

Admiral Owens does not address the potential role of modeling and simulation in exploiting battlespace awareness but he hints in this direction with his comments about C^4I, precision-guided munitions (PGMs), and intelligence, surveillance and reconnaissance (ISR) system programs:

> ... we tend to deal with these programs separately, it is difficult to recognize that together they posit—qualitatively—a quite different military potential. The interactions and synergism of these systems constitute something new and very important. What is happening is driven in part by broad conceptual architectures—and part by serendipity: It is the creation of a new system of systems.[*]

Admiral Owens' "system of systems" is depicted in Figure 1.[*]

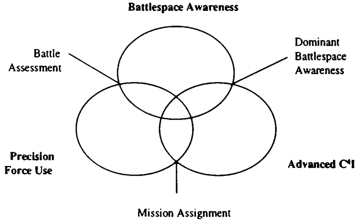


Figure 1: Admiral Owens' System of Systems

III. Challenges Of Modeling And Simulating DBA

There are a number of challenges to successfully modeling Dominant Battlespace Awareness. The world has clearly changed in the over the last six years. Not only have political events of unimagined import occurred, but we have had glimpses of a change in the nature of warfare as we know it. The Gulf War has been heralded by some as the first war of a new era—and by others—as the last of an era. What is clear is that warfare will never be the same as it was before the Gulf War, whether you view it as an opening act or a finale.

Over the period of the Cold War we had developed some fairly extensive modeling and simulation capabilities. There were recognized shortcomings: Most of the models were aimed at simulating attrition-style warfare. There was no consensus as to what were effective measures of effectiveness (MOEs). And models rarely, if ever, depicted, interactively, the capabilities of more than one system. There were few, if any attempts to model asymmetric warfare (warfare other than force-on-force).

Today we are seeing the emergence of new forms of warfare. Some suggest that these are: information warfare, precision strike, dominating maneuver, and space warfare. Figure 2 depicts the potential Revolution in Military Affairs (RMA) that Science Applications International Corporation (SAIC)'s Strategic Assessment Center (SAC) believes may be occurring at the intersection of these warfare areas. None of today's models even attempt to model these warfare areas, no less, their intersection.

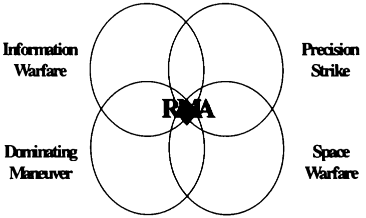


Figure 2: Revolution in Military Affairs

We are entering an era of emerging warfare areas and emerging ways of waging war. As Alvin and Heidi Toffler suggest in War and Anti-War, we may be entering a post-industrial, or information age.[†] In areas where new forms of warfare combine, exists the potential for new forms of warfare. We may be seeing the emergence of "derivatives warfare"— where adversaries leverage precision actions, both through physical destruction (precision strike) and information warfare, against limited targets for limited gains, that, in combination, totally overwhelm the adversary's ability to fight effectively—or never even reach the thresholds normally considered necessary to elicit/justify a military response. In an era where adversaries may leverage both military and non-military forces and pressures to achieve limited, incremental gains through operations other than war, it will be necessary to respond quickly and in a limited manner, while the circumstances are still right and perishable national will is energized. This is tough to do—and even tougher to model.

Traditional force-on-force models have generally been used in the formulation of target lists and to depict the effects of attrition warfare across one dimension of a single battlespace. If a Revolution in Military Affairs is in fact occurring we may be witnessing the emergence of new forms of warfare. For example, when precision strike and dominant battlefield awareness are combined true non-linear warfare may be possible. In a world where the two are married, battle fronts, as we know them, may lose their significance, and once powerful, threatening targets may, at best, become irrelevant, or merely just inconsequential, because of their avoidable location, relative immobility or slow speed, or relatively shorter-range systems. Today's models cannot depict this. This combination of forces—and the synergies possible between the other RMA warfare areas—may portend different types of warfare on different types of battlefields against different types of targets.

IV. M&S Shortcomings

The critical shortcomings in using today's models and simulations to address the RMA, in general, and DBA in particular, are many. These failings include the effects of:

- C4I: their role in offensive and defensive operations, the result of a degradation in C4I capability on military operations, etc.

- Time: what does an increasing tempo and/or simultaneity of effects mean to the defeat of an enemy force?

- Maneuver: positioning forces in such a manner so as to gain a disproportionate effect (i.e., we still cannot simulate a "blitzkrieg")

- Weapons of Mass Destruction (WMD): little understanding of the operational effect of use on forces (i.e., performance in protective gear over days, weeks, or months) and C^4I (e.g., EMP effects and population)

- Cascading effects: very hard to gauge second order effects because of the tremendous integration of future C^4I systems; time delays in the target area may have unexpected and important implications

- Intangibles: we still cannot model the value of morale, leadership, training, or the "Fog of War"

- Change: the complexities of RMA operations is likely to continue to increase faster than the modeling and simulation community can quantify them

- Precision warfare: models are not currently capable of accommodating the increasing levels of precision central to future conflicts on a theater scale (e.g., hex-based models have difficulty reconciling the trade-offs between large degrees of uncertainty within a given area (i.e., 10 km) and overloading the system with detail (i.e., 72 m), making it all but impossible to generate useful results in a timely manner)

V. Integration Of RMA Warfare Areas And Dominant Battlespace Awareness

Figure 3 depicts the Engagement Cycle and its Component Elements. Figure 4 illustrates the interrelationships between the Revolution in Military Affairs warfare areas, their relationship to the engagement cycle and the elements of dominant battlespace awareness that must be adequately modeled and/or simulated to achieve and evaluate truly integrated warfare. The systems of systems depicted here all must play a role in Admiral Owens' *Emerging System of System*. To fully utilize DBA and achieve integrated warfare will require the seamless linking of all of the RMA warfare areas, as well as, basing and logistics. Modeling and simulation are a must for developing and employing this integrated form of warfare.

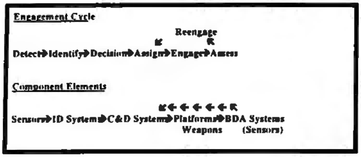

Engagement Cycle
Reengage
Detect→Identify→Decision→Assign→Engage→Assess
Component Elements
Sensors→ID System→C&D System→Platforms→BDA Systems
Weapons
(Sensors)

Figure 3: Engagement Cycle and Components

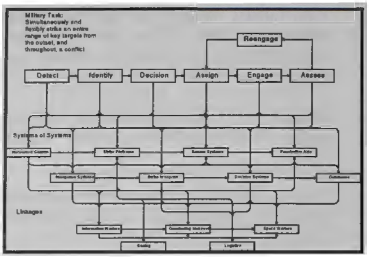


Figure 4: Integrated Warfare

In the near term we must live with the separate systems we have today. But we can take steps, using M&S, to test and tune future integration. The first steps are to:

- Develop models and measures of effectiveness to address the new warfare areas

- Start to use modeling linkages to tie together the disparate elements that make up our non-system of systems, to begin to develop the non-existent interchanges that take advantage of existing potential synergies

- Use models and simulations to develop "wrappers" to encapsulate unruly and uncooperative system elements so that they can interact with other elements in the most opportune manner

- Use simulations as "fillers" or "placeholders" for not-yet-developed system elements, to take the fullest possible advantage of asynchronous system developments, allowing system elements to come "on line" when they are ready, rather than waiting for entire system(s) maturation

- Develop models and simulations that aid in the actual fusion of battlespace awareness inputs, acting as translators, interpreters, facilitators, substitutions or imitators/metaphors, where necessary. Figure 5 depicts the role that M&S could play in developing Admiral Owens' emerging system of systems

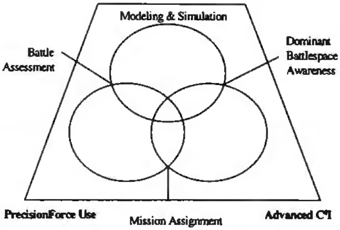


Figure 5: M&S and System of Systems

- Examine the potential of software "intelligent agents" and "spiders" to search out important data and link critical sites[‡]

- Investigate the potential of Expert and Knowledge-Based Systems, artificial intelligence, and fuzzy logic to assist in automating the decisionmaking process

VI. Modeling And Simulation, Information Presentation And Assisted Decisionmaking

Modeling and Simulation can play a major role in information presentation and assisted decisionmaking. Three Dimensional (3-D) Holographic display systems, supported by color-ized large screen displays could be used to provide a virtual reality environments where commanders and staffs would have the best possible "relative" or "virtual" picture. Commanders would no longer have to go through the mental gymnastics required to convert "real" or "true" data (latitude/longitude, bearing/range, course/speed) into the "virtual" picture that commanders already generate in their minds. The easier the commander can interpret "the picture," the quicker he, or she, can make a decision.

It is time to explore the opportunities that "virtual reality" offers us. It is conceivable that a "virtual" picture of the battlespace may now be more accurate and usable than a real, visual one. Fused information and intelligence presented in the form of 3-D images that mimic the "real McCoy" may be a more appropriate target to shoot at. Visual 3-D depictions of weapons pairings to visually identifiable 3-D targets that warriors can examine from all angles present some very interesting implications for the speed and accuracy with which warfare can be conducted.

It may also be possible to depict and model all forms of "lines of communications" on the "information terrain." Topographically replacing physical geography with "infography" may lead to considerable changes in the way we conduct warfare. Modeled information "peaks" and "ridges" may become the "centers of gravity" of future warfare. We may be faced with the opportunity/necessity to view targeting from a much different viewpoint—where information intersections become as important as nodes or command bunkers—both from an offensive and defensive viewpoint. Many information warfare "targets" do not exist in a physical sense, and therefore can only be modeled or simulated, in order to be presented in a visual and comprehensible manner.

Modeling- and simulation-assisted warfare could free-up the commander and their staff to deal with only the hard choices, the ones where the system could not solve the problem. Under such a system, the fusing of multiple source intelligence, information (and BDA) to generate a high quality system target that exceeds the accuracy of the any single sensor. This parallel decision-making process would free our military forces from the current organic sensor-organic weapon "lock" (sensor and weapon hard-wired to each other on the same platform). It would permit, in a sense, the parallel pairing of any sensor(s) with any weapon(s).

By making parallel decisions, instead of today's serial ones (one-at-a-time, one-after-another) a must faster rate of warfare is possible. By further inputting mission priorities, rules of coordination and engagement, and an "acceptable degree of difficulty" into the system, the commander can set a required "confidence" level that must be achieved before an engagement is ordered. In a touchy face-off with a nuclear-armed peer competitor, a commander could require a confidence level of 100% for every engagement.

Against a third-world niche competitor where quick decisive action is required, the commander may be willing to lower the required average confidence level, to say, 55-60%, with a low "degree of difficulty" threshold. Using modeling, simulation and assisted decisionmaking could allow the consolidation of decisionmaking time to the point in the decisionmaking process where it makes the greatest difference—just prior to the engagement—taking advantage of automation to make all preceding decisions in parallel engagement processes.

In a typical example using an automated parallel decision-making process, and command by negation (CBN), the decisionmaker has two full minutes to decide whether three engagements should proceed, over three ten-second decision times in a sample serial process.

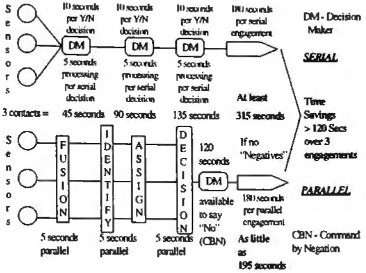

Sensors
10 seconds per Y/N decision
10 seconds per Y/N decision
10 seconds per Y/N decision
180 seconds per serial engagement
DM - Decision Maker
DM
DM
DM
SERIAL
5 seconds processing per serial decision
5 seconds processing per serial decision
5 seconds processing per serial decision
At least
3 contacts =
45 seconds
90 seconds
135 seconds
315 seconds
Time Savings > 120 Secs over 3 engagements
Sensors
FUSION
IDENTIFY
ASSIGN
DECISION
120 seconds
If no "Negatives"
DM
PARALLEL
available to say "No" (CBN)
180 seconds per parallel engagement
5 seconds parallel
5 seconds parallel
5 seconds parallel
As little as 195 seconds
CBN - Command by Negation

Figure 6: Serial Vs. Parallel Decisionmaking

Information technology no longer requires that the "sensor" and "shooter" be on the same platform, and, in some cases, even in the same geographic or operational arena, or theater. It also allows for real-time sensing and shooting. National, or remote, sensors may have the best "picture" from which to conduct an engagement. A "local" sensor may be paired with a theater or national weapon. An Army Ranger armed with a Tomahawk cruise missile or a stealth-delivered bomb is a powerful weapon. An Aegis cruiser or destroyer's main battery is not its onboard missiles, it is a Navy F-14, a Navy or Marine F/A-18 or an Air Force Fighter. A national sensor may be similarly paired with a local weapon, see Figure 7.

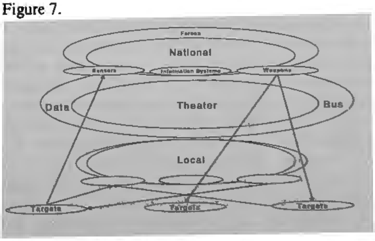


Figure 7: Virtual "Sensor to Shooter" Organization

Modeling and simulation can play another important role in data fusion and information presentation. One of the expectations in the early phases of dominant battlespace awareness is that systems and decisionmakers will be overwhelmed by data and decisions. Modeling and simulation can be used to simplify the problem. Today, all decisions and all data collections and presentations are given, as a minimum, the same basic allotment of time; be it for search, identification, tracking, or targeting. Not all presentations and decisions require even this same basic amount of time. Simulations can be used to monitor and model the priorities and engagements of non-critical targets, on a periodic basis, not inputting them into the human or automated/assisted decisionmaking queues until absolutely necessary. Not all targets deserve the same amount of decisionmaking time or priority. M&S can be used to automate portions of the decisionmaking process, reserving and preserving human involvement until it is absolutely necessary. It can even be used to fully automate some entire engagement processes.

Given dominant battlespace awareness, command-by-negation, new targeting philosophies, and existing "automatic" engagement systems, it is possible to use modeling and simulation to dramatically reduce the possibility of "blue-on-blue" engagements, without endangering our forces. Many systems, such as the Aegis Weapons System, were designed to be able to automatically engage massive Soviet raids. These "last ditch" features were meant for engagements when no human could possibly make and implement decisions quickly enough to defeat a massed raid. The parameters that threats had to meet to be automatically engaged were normally very exact. This was done for two reasons: to prevent the likelihood of an undesired engagement and to increase the likelihood of a successful engagement if the tight parameters were met. These automatic systems may have a new purpose, they can be used very effectively in an increasingly information-aware world. They are microcosms of reconnaissance-strike-targeting architectures, with a defensive flavor. Many of these systems can, and do, take inputs from identification systems, such as IFF (Identification, Friend or Foe). IFF and other safeguards help to prevent their accidental use against non-desired targets. Use of these systems, and the other concepts presented here, by the USS VINCENNES in 1989 and the Airborne Warning Control System (AWACS) aircraft and fighters over Northern Iraq in 1994, might have prevented two avoidable disasters. The addition of continuous, or event-triggered modeling and simulation, as another "check and balance" in the system, could allow these systems to perform a revised, and critically important, function.

VII. Conclusions

Modeling and Simulation may be the key to converting C^4I into Dominant Battlespace Awareness. Fused intelligence, linked with prepackaged digitized terrain, may finally give commanders and warriors a "complete" picture of the battlespace that far exceeds that possible from any single source. Modeling and Simulation may also be the missing link in successfully automating decisionmaking processes. Modeling and Simulation can be used to greatly enhance what we do know, but cannot perceive, about targets with uncorrelated information; and can help to clearly to identify the "real unknowns"—those targets about which we know nothing.

This use of Modeling and Simulation will affect strike and mission planning, defensive operations, targeting, logistics, operations and command and control. It could conceivably contribute to the development of a new type of warfare for the twenty-first century and a true Revolution in Military Affairs (RMA).

VIII. References

[*]Admiral William A. Owens, USN, "The Emerging System of Systems," U.S. Naval Institute PROCEEDINGS, May 1995, p. 37.

[†]Alvin and Heidi Toffler, War and Anti-War, (Boston: Little, Brown & Company, 1993)

[‡]Philip E. Ross and Nikhil Hutheesing, "Along Came the Spiders," Forbes, October 23, 1995, pp. 210-216.

AIAA-96-3474-CP

ANALYSIS AND COMPARISON OF THE MOTION SIMULATION CAPABILITIES OF THREE-DEGREE-OF-FREEDOM FLIGHT SIMULATORS

Nicolas A. Pouliot*, Meyer A. Nahon† and Clément M. Gosselin*

*Département de Génie Mécanique, Univ. Laval, Québec, Qc, Canada, G1K 7P4
†Department of Mechanical Eng., Univ. of Victoria, Victoria, B.C., Canada, V8W 3P6
email: pouliot@gmc.ulaval.ca, mnahon@sirius.uvic.ca, gosselin@gmc.ulaval.ca

Abstract

This paper presents the results of a preliminary study aimed at determining the simulation realism which could be achieved using reduced degree of freedom flight simulator motion bases. More specifically, the quality of motion produced by two different three degree of freedom (3-DOF) platforms was compared to that produced by a standard 6-DOF Stewart platform. The 3-DOF motion bases investigated include a spherical mechanism which allows only rotational motions, as well as a motion base capable of heave, pitch and roll motions. To compare the different motion bases, four characteristic maneuvers were simulated using a non-linear model of a Boeing 747. The aircraft motions were then simulated on nine different combinations of virtual motion platforms and motion base drive algorithms. The motion cues (specific forces and angular velocities) produced in this manner where then graphically compared. The analysis revealed that, in most cases, a 3-DOF simulator is capable of producing motion simulation quality comparable to that produced by a 6-DOF Stewart platform.

1 Introduction

The six-degree-of-freedom (6-DOF) Stewart platform is undoubtedly the most popular motion base for commercial flight simulators[1] However, it is a complex and expensive mechanism which is best suited to full training simulators. Recently, there has been renewed interest in low-cost partial-training simulators which could be used for early pilot training, prior to the use of full-training simulators. As well, the advent of high-performance Image Generation (IG) systems[2] has tended to decrease the relative importance of motion generation, thereby raising the issue of whether lower cost motion devices might be appropriate even for full-training simulators. Finally, there has recently been heightened interest in lower cost simulators for applications to road vehicle simulation and entertainment.

Relatively little work exists in the area of the design of reduced-dof motion bases or in the evaluation of the quality of motion sensations which could be produced by these devices[3,4,5]

The purpose of this paper is therefore to analyze the simulation realism which can be achieved using alternative motion base mechanisms, with only 3 degrees of freedom. It is conjectured that this reduced motion capability might provide a cheaper alternative to existing designs (simplicity, reduced cost of manufacturing and operation) while still producing a good quality of motion simulation for large transport aircraft. In all, nine combinations of motion base architecture and drive algorithms were evaluated using four different aircraft maneuvers. The evaluation maneuvers used in this study were chosen to ensure that a limited number of maneuvers would provide a broad range of motions, including both low and high frequency accelerations in all degrees of freedom.

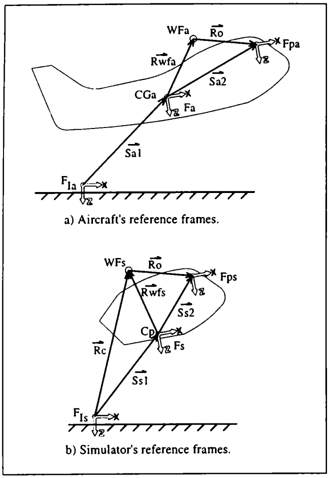
2 Background Material

2.1 Type of aircraft and modeling

The present work focusses on the simulation of commercial airliners, and the Boeing 747 has been chosen as a typical example. This emphasis is due to the fact that, due to their large inertia, the natural motion of these aircrafts lends itself well to motion simulations which are principally composed of low-frequency rotations (including tilt-coordination). Thus, high-frequency translational accelerations such as those required for the simulation of military aircrafts would tend to be poorly simulated by the alternative platforms considered here. A complete nonlinear model of the Boeing 747[6,7]is used in the present work to predict aircraft motion in response to a range of pilot inputs and randomly generated turbulence while flying at low altitude.

2.2 Reference frames

The reference frames associated with the aircraft are shown in Fig. 1a. Let F_{Ia} be the inertial reference frame, fixed to the ground. By convention, the Z axis points vertically downwards and the X axis is parallel to the active runway. A second reference frame F_a is attached to the aircraft and has its origin at the center of mass of the aircraft $\mathbf{CG_a}$. The X axis of F_a points longitudinally forward while the Z axis points downward with respect to the aircraft. As a result, the Y axis points out the right wing. Furthermore, vector $\vec{S}_{a1}$ is defined as the vector connecting the origin of frame F_{Ia} to the origin of frame F_a, as illustrated in Fig. 1a. Finally, Euler angles $\vec{\beta}_a^T = (\phi, \theta, \psi)$ are used to specify the relative orientation of frame F_a with respect to inertial frame (F_{Ia}). Hence, the inertial position of a point of the aircraft whose position vector with respect to frame F_a is given by $[\vec{P}]_{F_a}$ can be written as



a) Aircraft's reference frames.

b) Simulator's reference frames.

Figure 1: Reference frames.

$$[\vec{P}]_{F_{Ia}} = [\vec{S}_{a1}]_{F_{Ia}} + \mathbf{Q}_a[\vec{P}]_{F_a} \tag{1}$$

where

$$\mathbf{Q}_a = \begin{bmatrix} c\theta c\psi & s\phi s\theta c\psi - c\phi s\psi & c\phi s\theta c\psi - s\phi s\psi \\ c\theta s\psi & s\phi s\theta s\psi - c\phi c\psi & c\phi s\theta s\psi - s\phi c\psi \\ -s\theta & s\phi c\theta & c\phi c\theta \end{bmatrix} \tag{2}$$

and where c stands for cos and s for sin. The relationship between the time derivatives of the Euler angles and the angular velocity vector of the aircraft, $\vec{\omega}_a$, is then written as

$$\vec{\omega}_a = \mathbf{R}_a \dot{\vec{\beta}}_a \quad \mathbf{R}_a = \begin{bmatrix} 1 & 0 & -s\theta \\ 0 & c\phi & s\phi c\theta \\ 0 & -s\phi & c\phi c\theta \end{bmatrix} \tag{3}$$

A third reference frame, denoted F_{pa}, having the same orientation as F_a is also defined, with its origin at the center of the pilot's head. The vector connecting the origins of these two frames, $\vec{S}_{a2}$, is illustrated in Fig. 1a. In the present study, this vector has been assigned the value $\vec{S}_{a2}^T = \{26.2, -0.465, -3.4\}$ m, which approximately represents the pilot's head location in a Boeing 747.

The reference frames associated with the simulator's motion base, are shown in Fig. 1b. An inertial reference frame (F_{Is}) is fixed to ground, directly below the center of the simulator's motion base, when the simulator is in its neutral position. Moreover, frames F_s and F_{ps} are attached to the moving platform of the simulator with their origins respectively located at the geometric center of the simulator's moving platform **CP** and at the center of pilot's head **PH**. The latter frames have the same orientation, which is given by the simulator's Euler angles, using the convention defined for the aircraft. Finally, vector $\vec{S}_{s1}$ is defined as the vector connecting the origin of frame F_{Is} to the origin of frame F_s and vector $\vec{S}_{s2}$ is defined as the vector connecting the origin of frame F_s to the origin of frame F_{ps}. Other points and vectors shown in Fig. 1b will be defined later.

2.3 Washout filter algorithm

The purpose of defining a reference frame with its origin at the pilot's head is to allow the determination of the angular velocities $(\vec{\omega})$ and specific forces $(\vec{f})$ to which the pilot is subjected. It is generally accepted that these cues plays a dominant role in human motion sensing activities[8]

It is recalled that the specific force is a vectorial quantity defined as the difference between the translational acceleration vector $(\vec{a})$ and the vector of gravitational acceleration ($\vec{g}$), i.e., $\vec{f} = \vec{a} - \vec{g}$. Hence, a body at rest under the action of the earth's gravitational field is said to be experiencing an upward specific force of $9.81m/s^2$, i.e. in the negative direction of the Z axis defined above.

Since the flight simulator motion base has a limited range of motion, it is not possible to exactly reproduce the angular velocities and specific forces experienced in the aircraft. Hence, motion generation algorithms (a.k.a. 'washout filters') have been developed to generate the best motion commands within the constraints of the motion system. The principles of the washout algorithm used in the present work will now be reviewed.

One of the most demanding tasks for a motion base with limited motion travel is the reproduction of low frequency translational accelerations. For instance, a constant acceleration of $1m/s^2$ along the X axis sustained for 5 seconds would require a travel of 12.5 meters. However it is also possible to slowly tilt the simulator about its Y axis at an angle of approximately 6 degrees, while the visual display continues showing horizontal flight conditions. The pilot would then experience a specific force component along the X axis of $9.81 \cdot \sin(6°) = 1m/s^2$. Although the Z component of the specific force experienced by the pilot would then be reduced to $9.75m/s^2$, this change should not be noticeable. The effectiveness of this process, known as *tilt coordination*, to simulate low frequency translational accelerations, leads to an increased importance of the rotational channels in relation to the translational channels.

The washout filter (WF), which includes the above-mentioned tilt-coordination, receives as inputs the angular velocities and specific forces sampled at a particular location in the aircraft – referred to as the washout location and denoted $\mathbf{WF_a}$ and $\mathbf{WF_s}$ on Fig. 1 – and outputs the optimum displacement vector ($\vec{S}_{S1}$) and the Euler angles ($\vec{\beta}_s$) that the simulator's cockpit should assume. The WF also continually tries to return the motion platform to the central location of its range of motion (the neutral point).

There exist several variations of WF algorithms[9] but its classical form has been used here due to its widespread use in commercial simulators. A complete description of this algorithm can be found in Nahon *et al.*, 1992[9] while Fig. 2 illustrates the algorithm, as used in the present work. However, the dashed-line block denoted as *HF emulator* is an addition developed for the purposes of the present work only, and will be described in a subsequent section. Consequently, boxes 13 to 17 should be non-existent in the conventional classical WF algorithm.

As can be seen in Fig. 2, the washout algorithm is broadly divided into three parallel channels of motion. The upper channel (translational), consisting of boxes 1 to 4, scales the input specific forces ($\vec{f}_{aa}$), typically by scale factor of 0.5, transforms the resulting vector into the inertial reference frame and then adds the gravity vector. This produces the acceleration vector ($\vec{a}_c$) which must be filtered to extract only its high frequency component ($\vec{a}_{s1}$) in order to assure that the simulator will remain near its neutral point. Finally, ($\vec{a}_{s1}$) is integrated twice to obtain the required displacements of the platform ($\vec{S}_{s1}$). The lowest channel (rotational), consisting of boxes 7 to 10, is similar to the translational channel but acts upon angular velocities. Hence box 10 only carries out a single integration, thus producing a set of Euler angles ($\vec{\beta}_h$). The central channel, consisting of boxes 5 and 6, represents tilt-coordination. Box 5 receives the scaled specific forces and extracts their low frequency components which are then transformed by box 6 into tilt angles ($\vec{\beta_{TC}}$). Box 6 also includes a rate limiter to ensure that the tilt-coordination will occur slowly enough to keep the effect realistic. The sum of $\vec{\beta_h}$ and $\vec{\beta_{TC}}$ provides $\vec{\beta_s}$, the orientation of the simulator, which is then used to compute the matrices $\mathbf{Q_s}$ and $\mathbf{R_s}$.

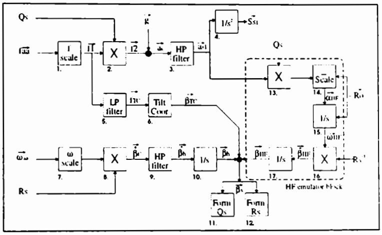


Figure 2: Washout filter algorithm layout.

3 Methodology

In this work, aircraft simulation software developed at the University of Toronto Institue for Aerospace Studies (UTIAS) to drive a full scale six-degree-of-freedom simulator[10]was used off-line on a workstation. Some of the algorithms included in the software were modified in order to simulate the three-degree-of-freedom motion bases evaluated in this work. Indeed, it is possible to artificially reduce the simulator's number of DOF by inhibiting, for instance, all the translations while keeping the rotational motion, thus emulating the behavior of a three-DOF spherical simulator.

Figure 3 summarizes the various steps which were required in order to produce the results. The primary inputs consisted of two standard text files. The first file gives the initial conditions which are necessary in order to solve the differential equations of flight. The second file is a complete time history of the aircraft's control inputs (elevator deflection, throttle position, wheel angle, etc.) for the entire simulated maneuver and with a sampling frequency of 20 Hz. The aircraft model, with the help of three major modules (engine model, turbulence generation and ground model) then outputs the simulated flight history to a file. The accelerations and angular velocities thus obtained are then fed directly into the vestibular model and into the WF algorithm. The vestibular model, described in detail in Reid and Nahon, 1985[10], is use to determine wether conclusions based on pilot sensations would differ appreciably from ones based on motion time histories. As depicted in Fig. 3, the WF transforms the flight history into the simulator motion time history, according to which degrees-of-freedom are currently available.

Finally, the simulator specific forces and angular velocities are fed into the vestibular model, which processes both the aircraft and simulator flight histories to produce the pilot sensations time-histories.

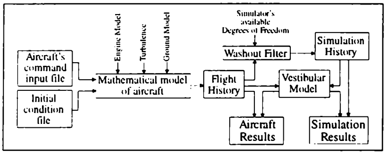


Figure 3: Layout of the methodology.

The methodology described above produces results which can be used to compare the quality of the motion simulation obtained with a 3-DOF simulator with that obtained with a 6-DOF simulator. Additionally, the results obtained with both types of simulator can also be compared with the real flying experience.

3.1 Evaluation maneuvers

Since it would not be possible to test an exhaustive set of aircraft flying sequences, a limited number of maneuvers had to be selected. Four characteristic maneuvers were used to perform the comparison between the simulators. These maneuvers were chosen to exercise the full range of simulator DOFs.

The first maneuver started with the initial conditions referred to as *runway conditions* while the others started with the *cruise conditions*. These initial conditions are summarized in Table 1. A brief description of each of the four maneuvers follows.

Table 1: Initial conditions files content.

I.C. FILE:	runway	cruise
Speed (m/s)	50.0	212.5
Altitude (m)	4.67	6280
Engine EPR	1.02	1.10

Take-Off Maneuvre (TOM): This maneuver begins at a speed of 50 m/s on the runway with the throttle lever at 15% of maximum power. After three seconds, the throttle is abruptly set to 100% and the aircraft accelerates. At t=11s, when rotation speed is reached, the nose is lifted and soon after, the aircraft takes off at a steady pitch angle of 10 degrees. Finally, six seconds later, a sudden failure occurs on the right outboard engine and the pilot eases the elevator to keep the plane level. The maneuver ends at t=25s.

Turn-Entries Maneuvre (TEM): In this 40 seconds maneuver the aircraft, flying at cruise condition, first rolls right, then left, and then right again. The maximum bank angle in each roll is approximately 40°. The pilot adjusts the elevator to keep the aircraft's altitude almost constant and the throttle lever is set at 55% of maximum power for the entire maneuver.

Throttle Impulse Maneuvre (TIM): The pilot, initially flying at cruise condition with 55% power, suddenly increases the throttle setting to maximum power while adjusting the elevator to keep the aicraft's altitude constant. Seventeen seconds later the throttle is set to idle until the end of the maneuver, at time 40 seconds.

Turbulence Area Maneuvre (TAM): This maneuver begins with the aircraft trimmed for cruise condition, when a patch of turbulence is encountered. The pilots makes no corrective control inputs. The controls therefore remain in the same state as if the plane were cruising in still air for a duration of 25 s.

3.2 Location of washout:

The original washout filter software taken from Reid and Nahon, 1985[10],always performed the washout calculations using values of specific forces existing at the location in the aircraft corresponding to the centroid of the motion base. However, the present work also investigated alternative washout locations. As shown in Fig. 1, the location of WF_a (and WF_s), at which washout is performed, is defined with respect to the pilot's head by means of the vector $\vec{Ro}$. Preliminary tests indicated that it was imperative that the vector $\vec{Ro}$ be identical in both the aircraft and the simulator (i.e. that WF_a and WF_s be in the same location relative to the pilot's head). Attempts to use other configurations invariably led to the generation of unwanted spurious translational accelerations.

However, in order to compute the hydraulic cylinder lengths, the kinematics transformation software requires the position of the centroid of the motion base, denoted by **Cp** on Fig. 1, to be specified. To accomplish this, even when washout might not be performed at **Cp**, vector $\vec{Rwf_s}$, must be calculated according to $\vec{Rwf_s} = \vec{S}_{s2} - \vec{Ro}$ where $\vec{S}_{s2}^T = (-0.02, -0.465, -1.783)m$, expressed in frame F_s. This value for $\vec{S}_{s2}$ represents the pilot's head location relative to the centroid of the moving platform for the UTIAS flight simulator.

Three locations for the washout filter were thus evaluated in the present work:

@PH: The pilot's head. In this case, $\vec{Ro}^T = (0,0,0)$ and $\vec{Rwf}_s^T = \vec{S}_{s2}^T$. The benefit of this scenario is that no spurious translational accelerations are generated at the pilot's head.
@CP: The centroid of the moving platform. In this case, $\vec{Ro}^T = \vec{S}_{s2}^T$ and $\vec{Rwf}_s^T = (0,0,0)$. This is the usual convention, as used in Reid and Nahon, 1985[10].
@CG: The center of gravity of the moving platform and cockpit. For the UTIAS simulator, this location has been approximated as $\vec{Ro}^T = (-0.02, -0.465, -0.483)m$. The benefit of this location is that it tends to minimize the dynamic actuation forces, since off-center rotations are not generated.

3.3 Alternative motion base architectures:

In the present work, the three different motion base architectures were simulated and compared. The last two use a particular combination of three degrees of freedom which could be emulated using a real six degree-of-freedom simulator.

STW: The first architecture is the conventional Stewart platform which possesses a full six degrees of freedom. It can translate along all three orthogonal directions independently, as well as rotate about these same axes. A schematic representation of this motion base is shown in the uppermost frame of Fig. 4. At a given time step, the washout filter algorithm produces $\vec{S}_{s1}$ and the orientation matrix $\mathbf{Q}_s$.

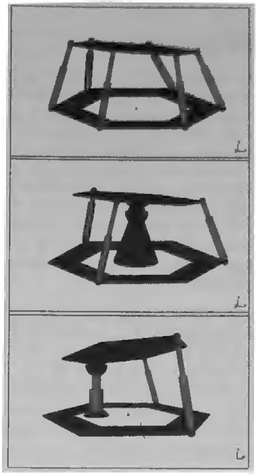


Figure 4: Different Motion Base Architectures.

SPH: The second architecture considered has only three degrees of freedom, consisting of rotations about three mutually perpendicular axes. The middle frame of Fig. 4 shows a representation of what the motion base might look like. As it was highlighted in the discussion of tilt-coordination, the rotational motion channels are particularly important in the simulation of large transport aircraft. Since the platform is only capable of rotations, the only useful ouput from the washout algorithm is the orientation matrix $\mathbf{Q}_s$. The linear displacements are systematically set to zero and the location at which washout is performed ($\mathbf{WF}_s$) will then not be allowed to move with repect to the ground.

HPR: The third architecture also has three degrees of freedom but this time including translation along the vertical axis (heave). The other two degrees of freedom consist of rotations about the two horizontal axes, namely the pitch (around the Y axis) and roll (around X axis). The notation HPR therefore stands for Heave-Pitch-Roll. The last frame of Fig. 4 shows a representation of what this motion base might look like. The point $\mathbf{WF}_s$ will then move along a vertical axis while the two rotations are performed about this point. In this case, the useful output of the washout filter are the Z component of $\vec{S}_{S1}$ and the roll and pitch components of $\vec{\beta}_s$.

3.4 A HF emulator

In order to compensate for the loss of translational motion in the 3-DOF **SPH** and **HPR** architectures, a further modification of the washout algorithm, called a 'high frequency (HF) emulator', was evaluated. Its purpose is to reproduce some of the high frequency translational accelerations by commanding an appropriate angular acceleration assuming that the pilot's head is displaced from the center of rotation by a non-zero vector $\vec{Ro}$. The HF emulator, consisting of boxes 13 to 17 in Fig. 3, was thus implemented into the washout filter algorithm, and is described below.

First, assume that the high frequency acceleration which must be emulated is denoted by $\vec{a}_{s1}$. This acceleration can be transformed into the simulator reference frame according to $\vec{a} = \mathbf{Q}_s^T \vec{a}_{s1}$. Since an angular acceleration can only produce linear accelerations which are perpendicular to $\vec{Ro}$, the following transformation is performed on $\vec{a}$ to remove the acceleration component along $\vec{Ro}$:

$$\vec{a}_\perp = \vec{a} - (\vec{a} \cdot \vec{Ro})\frac{\vec{Ro}}{||\vec{Ro}||^2} \tag{4}$$

The resulting vector now represents the component of the original high frequency acceleration which can be emulated. A corresponding angular acceleration can then be calculated according to:

$$\vec{\alpha}_{HF} = \frac{||\vec{a}_\perp||}{||\vec{Ro}||} \frac{(\vec{a}_\perp \times \vec{Ro})}{||(\vec{a}_\perp \times \vec{Ro})||} \tag{5}$$

That value can be integrated once to obtain $\vec{\omega}_{HF}$, then multiplied by the inverse of R_S matrix in order to get the derivative of the Euler angles and these are finally integrated, giving as a results a value for β_{HF}.

The HF emulator is automatically disabled when using the 6-DOF **STW** motion base. When the HF emulator is used with the 3-DOF **SPH** motion base, the vector $\vec{Ro}$ is set to (1.25, 0, 0), representing a center of rotation located behind the pilot, and allowing mainly Y and Z accelerations to be emulated. When the HF emulator is used with the 3-DOF **HPR** motion base, $\vec{Ro}$ is set to (0,0,-1.25) since high frequency vertical accelerations need not to be emulated (they can be produced directly by the motion base).

Table 2 summurizes the 9 different combinations of motion bases, washout filter locations and HF emulator that have been used in this study for each of the 4 maneuvers. Thirty-six sets of results where therefore generated and will be presented in the next section.

Table 2: Studied Architectures.

	@PH	@CP	@CG	+HF
STW(6-DOF)	•	•		
SPH(3-DOF)	•	•	•	•
HPR(3-DOF)		•	•	•

3.5 Conservative assumptions:

Additional assumptions were made to ensure that this study would remain conservative. The first was to assume that all three motion bases would have the same range of motion capabilities for those DOFs which were active. This assumption was dictated by the need to allow a future comparison to be performed on a standard 6-DOF motion base. It should be noted, by means of Table 3 from Reid and Nahon, 1988[11], that a 3-DOF motion base could be designed to have a much wider range of angular travel than the Stewart platform.

Table 3: STW motion base travel limits.

Linear		**Angular**	
X	±0.65 (m)	ϕ	±20.8(°)
Y	±0.59 (m)	θ	±21.3(°)
Z	±0.52 (m)	ψ	±20.0(°)
v_{max}	0.80 (m/s)	ω_{max}	34.4(°/s)
a_{max}	10.0 (m/s^2)	α_{max}	400(°/s^2)

A second conservative assumption was to use the same washout filter structure and coefficients for all motion bases (except for the HF emulator, where active). This washout filter had been designed to give good performance on the 6-DOF motion base described by Table 3. It should be noted, however, that the washout filter algorithm could likely be improved for operation specifically with the 3-DOF motion bases.

4 Results & Discussion

4.1 Performance indicators

The 36 sets of results mentioned in the preceding section, produced using different combinations of evaluation maneuver, motion base architecture, washout filter location and high frequency emulator, were evaluated in a number of ways. The initial results were generated as time-histories of different parameters. These included pilot control inputs; aircraft and simulator positions, velocities and accelerations; pilot vestibular model responses; and motion base actuator lengths. From these results, twelve graphs were then constructed: three for the components of the specific force $\vec{f}$; three for the the corresponding sensed specific forces $s\vec{f}$; three for the components of the angular velocity $\vec{\omega}$; and three for the sensed angular velocities $s\vec{\omega}$. On each graph, three curves were plotted: the first representing the aircraft's values, the second denoting the simulator's values, and the third giving the difference between the first two. In all cases, the conclusions based on the vestibular model results do not differ appreciably from ones based on motion time histories, and so only the latter will be shown here.

These results were then examined visually. However, since this type of evaluation would clearly be subjective, complementary objective comparison tools, called the *Performance Indicators* were also devised. The first of these indicators, $PI1$, is intended to yield a single numerical value which describes the average error in the motion variables experienced by the simulator pilot, while the second indicator, $PI2$, describes the average error in the rate of change of the same variables. The two indicators are described as $PIi = 100(A + B)$ where $i \in \{1, 2\}$ and where:

$$A = \frac{PIi_x + PIi_y + PIi_z}{3\,(a_{max})} \qquad (6)$$

$$B = \frac{PIi_{\omega x} + PIi_{\omega y} + PIi_{\omega z}}{3\,(\omega_{max})} \qquad (7)$$

Thus, each performance indicator is a weighted average of six component indicators, corresponding to the 6 motion channels. Moreover, since translational quantities have different unit dimensions than angular ones, all quantities were normalized by their corresponding upper limits a_{max} and ω_{max} from Table 3.

The component indicators which make up $PI1$ are defined as follows:

$$PI1_k = \frac{1}{\mathcal{F}T} \sum_{t=0}^{\mathcal{F}T} \left\| fk_t - fk_t^{sim} \right\| \qquad (8)$$

$$PI1_{\omega k} = \frac{1}{\mathcal{F}T} \sum_{t=0}^{\mathcal{F}T} \left\| \omega k_t - \omega k_t^{sim} \right\| \qquad (9)$$

where the subscript k represents the component X, Y or Z, being considered. Thus, the absolute value of the difference between a component specific force (or angular velocity) experienced in the aircraft and in the simulator is evaluated at each time step t. These are then summed over the complete maneuver, of duration T, and normalized by T and by the sampling frequency $\mathcal{F}$ (20 Hz)—that is, by the total number of elements in the summation.

The second performance indicator, $PI2$, aims to evaluate the difference between the *derivative* of the simulator and aircraft specific forces (and angular velocities), and is defined as:

$$PI2_k = \frac{1}{(\mathcal{F}T - 1)} \sum_{t=\Delta t}^{\mathcal{F}T} \left\| \frac{\Delta fk_t}{\Delta t} - \frac{\Delta fk_t^{sim}}{\Delta t} \right\| \qquad (10)$$

$$PI2_{\omega k} = \frac{1}{(\mathcal{F}T - 1)} \sum_{t=\Delta t}^{\mathcal{F}T} \left\| \frac{\Delta \omega k_t}{\Delta t} - \frac{\Delta \omega k_t^{sim}}{\Delta t} \right\| \qquad (11)$$

where, once again, the subscript k represents the component X, Y or Z, being considered. Thus, the derivative of the specific force $\Delta fk_t \Delta t$ (or of the angular velocity, $\Delta \omega k_t / \Delta t$) component is evaluated at time t using a first backward difference, where Δt is $1/\mathcal{F} = 0.05$ sec. The error in this quantity is taken as the absolute value of the difference between aircraft's and simulator's values. These are then summed over the duration of the maneuver and normalized by $(\mathcal{F}T - 1)$ since the summation now begins at $t = \Delta t$.

4.2 Graphical examples:

Seven plots are now presented to illustrate the results obtained. These are a condensed version of the plots previously mentioned, for the sake of compactness. Four curves now appear on each plot: a fine dotted line which shows the aircraft's behavior; a coarser dotted gray line which represents the behaviour of the 6-DOF Stewart platform; a black solid curve showing the behaviour of the 3-DOF platform under consideration; and finally, a dashed line which shows the difference between the two simulators (6-DOF and 3-DOF). No angular velocity plots have been presented in this paper since most of the 3-DOF architectures considered produce exactly the same motions as the 6-DOF platform: only the YAW axis of HPR motion bases and the HF emulator produced different angular time histories. Selected results are now shown for each of the four maneuvers, in turn.

Take-Off Maneuver (TOM): Early in this maneuver, the aircraft is subjected to a relatively large amplitude low frequency longitudinal acceleration as the aircraft accelerates down the runway. Later in the maneuver, at $t = 17s$ a high frequency motion is apparent when the engine failure occurs.

Figure 5 shows the X specific force produced by the 6-DOF STW@PH and the 3-DOF SPH@PH simulations. Tilt-coordination begins at $t = 3s$ in both cases. However, the tilt coordination rate limit (of 3°/sec) prevents both simulators from producing an X specific force buildup which is as rapid as in the aircraft. When the nosewheel is rotated, (at $t = 11s$), the tilt angle is further increased in both simulators to produce a larger X specific force. The circled region highlights the weakness of the SPH@PH in producing a high frequency acceleration: while the STW@PH manages to quickly reduce the X specific force in the same direction as the aircraft (though not to the same extent), the spherical motion base is constrained by the low frequency nature of its tilt-coordination. This difference is clearly visible in the ΔFx curve.

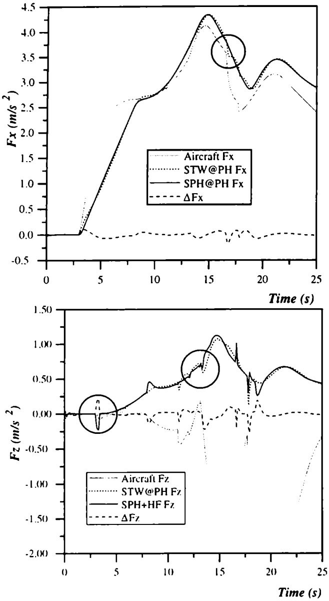


Figure 5: Fx and Fz in the TOM.

In a separate test with the same maneuver, the Z specific forces produced by a 6-DOF STW@PH motion base and by the 3-DOF SPH+HF motion base are also shown in Fig. 5. From this plot, it becomes clear that both simulators do a poor job of simulating this channel of motion. The reason, well known in flight simulation circles, is that tilt-coordination cannot be called upon to produce any vertical acceleration. Furthermore, in the attempt to generate a longitudinal specific force, any platform will invariably be constrained to generate positive (downward) values of Fz. The first circled region in this plot shows a sudden deviation of the SPH+HF from the STW which is due to the position of the pilot with respect to the center of rotation ($\vec{Ro}$). As a result of this displacement, when the tilt-coordination process begins, the pilot is subjected to an upward acceleration. The second circled region highlights an area where the HF emulator produces a beneficial effect. These and other graphical results for the TOM confirm that the spherical motion base appears almost as effective as the Stewart platform in this maneuver. In particular, the SPH@PH and SPH@CG architectures produce results which are especially close to the STW@PH curves. HPR architectures also performed relatively well, especially into the Z linear channel of motion. However, their weakness showed up when the Z angular channel is considered since no rotationnal motion is possible for them, even if these are required by the *engine-failure* simulation.

This analysis is confirmed, for TOM, by the values of the performance indices $PI1$ and $PI2$ as shown in Table 4. This table summarizes the objective analysis by giving for each of the maneuver the top-three 3-DOF architectures and their corresponding $PI1$ and $PI2$ values, as well as the best performing STW motion base and the worst architecture.

Table 4: Performance Indicators results for several architectures ($PI1$, $PI2$).

TOM	Best 3-DOF	SPH@PH(10.091, 0.190) SPH@CG(10.093, 0.192) SPH@CP(10.123, 0.214)
	Best 6-DOF	STW@PH(10.112, 0.200)
	Worst 3-DOF	HPR+HF(10.307, 0.219)
TEM	Best 3-DOF	SPH@PH(13.724, 0.299) SPH@CP(13.782, 0.273) SPH@CG(13.755, 0.293)
	Best 6-DOF	STW@CP(13.810, 0.276)
	Worst 3-DOF	HPR@CP(13.836, 0.277)
TIM	Best 3-DOF	SPH@PH(0.577, 2.035) SPH@CG(0.576, 2.073) HPR@PH(0.579, 2.051)
	Best 6-DOF	STW@PH(0.579, 2.071)
	Worst 3-DOF	SPH+HF(0.589, 2.241)
TAM	Best 3-DOF	SPH@CP(1.019, 0.403) HPR@CP(1.045, 0.369) HPR+HF(1.052, 0.362)
	Best 6-DOF	STW@CP(0.996, 0.353)
	Worst 3-DOF	SPH+HF(1.084, 0.388)

Turn-Entries Maneuver (TEM): This maneuver is the only one in which relatively large amplitude translational accelerations and angular velocities are generated in all three channels of motion for the entire duration of the maneuver. Figure 6 shows the Fx motion channel as it is simulated by STW@CP and HPR@CP. The near-zero ΔFx curve shows that the 3-DOF simulation is almost identical to the 6-DOF simulation. Recall that the angular velocities produced by both these simulators are identical except in yaw.

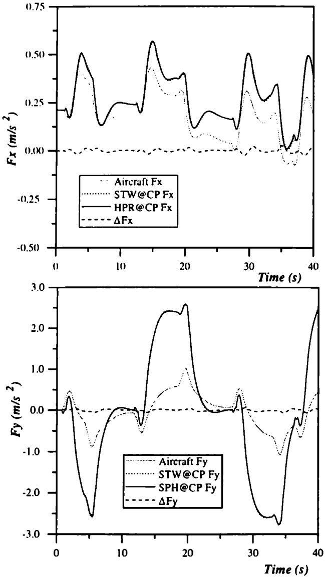


Figure 6: Fx and Fy in the TEM.

The similarity between the performance of a 3-DOF simulator and a 6-DOF simulator is again clearly visible from Fig. 6, where the SPH@CP architecture is used as a case in point. It is also clear from this figure that, during this maneuver, there is a large difference between the amplitude of aircraft's lateral specific force (Fy) and that experienced in both simulators. The aircraft tends to produce near-coordinated turns (i.e. the centripetal acceleration compensates lateral component of the gravity vector induced by the bank angle). The simulators, on the other hand, have no way of compensating for the lateral specific force produced by the bank angle. Once again, the Fz specific force was poorly reproduced in most cases, including the Stewart platform. In fact the simulator motion produced for this maneuver was nearly the same for all the 9 architectures considered, and hardly differed for the 3-DOF and 6 DOF simulators. Nevertheless, based on a visual inspection of their results, SPH@CP, SPH@CG and HPR@CP seemed to produce the best motion of the 3-DOF motion bases, while the three spherical architectures garnered the best results in terms of the performance indicators (see Table 4).

Throttle Impulse Maneuver (TIM): This third maneuver takes place completely in a vertical plane and all the cues that the pilot senses are in that plane. Moreover, this maneuver is the smoothest of the four studied here, and contains only low frequency motions. This therefore explains the excellent quality of simulation that was obtained with all the simulators considered. Figure 7 shows the motion produced by SPH@CG and STW@CP. It can be noted that all three curves are almost superposed for the entire maneuver.

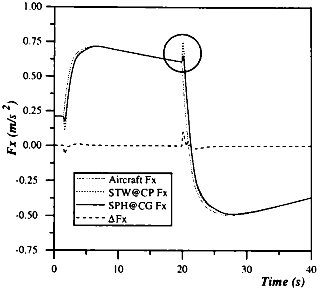


Figure 7: Fx in the TIM.

In fact this maneuver generates high frequency accelerations at only two times in its 40 seconds duration. These high frequency accelerations lead to a half-second delay which can be observed during the steep changes in Fx. However, the more serious effect of these high frequency accelerations are the sudden peaks (false cues) which are visible on the Fx curves produced by both simulators. Notice however that the false cues produced by the Stewart platform are almost three times worse than for the 3-DOF SPH motion base. In fact, the severity of these false cues is proportional to the length of $\vec{Ro}$ since they represent the translational acceleration induced by the angular acceleration caused by the onset of tilt coordination. Thus, no such peaks were observed in the @PH simulations where $||\vec{Ro}|| = 0$. These motion bases therefore tended to produce the best results for the TIM, as verified by the performance indicator values in the third column of Table 4.

Turbulence Area Maneuver (TAM): Airline pilots who train in simulators consider that the motion base's ability to adequately render HF accelerations such as turbulence or mechanical vibrations has a strong influence on their overall perception of realism. Hence this fourth maneuver is crucial to the evaluation of motion platforms with fewer than 6-DOF. However, since these accelerations are randomly generated and not due to pilot inputs, the general form of the simulated curve, such as its amplitude and frequency, tends to be more important than obtaining an exact replication of the aircraft's behavior. Figure 8 provides a good example of this since both simulators generate an adequate likeness of the aircraft's longitudinal specific force Fx, even though neither is able to reproduce it exactly.

However, as for all other maneuvers, the vertical specific force Fz is much more difficult to reproduce. As an example, consider Fig. 8 which shows the vertical specific force obtained with STW@CP and SPH+HF. In both cases, the motion produced in this degree of freedom has only about one third of the amplitude that was produced in the aircraft. In fact the amplitude obtained with the 3-DOF simulators without the HF emulator produced even smaller Fz value and, in the worst case, with SPH@PH, the vertical specific force was basically zero. Finally, considering all channels of motion, it appeared that SPH+HF and HPR+HF, as well as SPH@CP or SPH@CG were best able to simulate the TAM, and with a realism equivalent to the 6-DOF Stewart platform. Even though the performance indicator values seemed to be less relevant in the evaluation of TAM, they are nevertheless presented in Table 4.

5 Conclusion & Future Work

This paper has investigated the potential of a variety of 3-DOF motion platforms for the simulation of large transport aircraft motions. These platforms could be constructed and operated more economically than the present standard Stewart platform. In addition, such platforms might be appropriate for low cost procedures training simulators or for entertainment applications. Two basic architectures—one with three rotational DOFs, the other with heave, pitch and roll DOFs—were compared to the conventional Stewart platform. As well variations in the washout location and the addition of a 'high frequency emulator' were considered. In all, nine platform combinations were evaluated using four maneuvers. The comparison was mainly performed by visual inspection of graphical results but objective *Performance Indicators* have also been devised and used as a backup comparison tool.

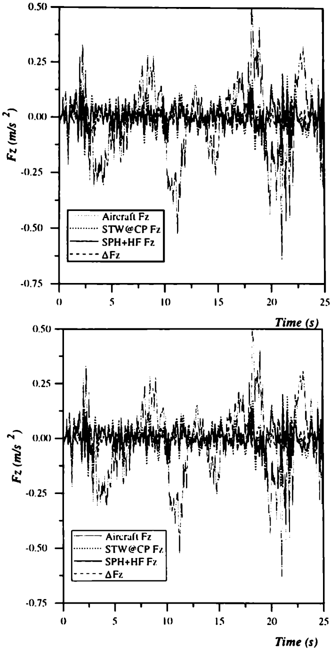


Figure 8: Fx and Fz in the TAM.

The analysis of the results revealed that, in most cases, a 3-DOF simulator is capable of producing a simulation quality comparable to that obtained with the 6-DOF Stewart platform. Based on the results, it is concluded that:

1. Three-DOF simulators represent a feasible alternative to 6-DOF motion systems since they can lead to a relatively good quality of simulation for large transport aircraft, such as the Boeing 747. Naturally, each of the motion degrees of freedom which was left unchanged (for example the angular velocities in the case of spherical mechanisms) produced identical motion histories. Maneuvers such

as Throttle Impulse (TIM) or Turn Entries (TEM) produced very similar results for all of the simulators since they involve low-frequency motions which can be well reproduced using primarily tilt-coordination.

2. The most effective, versatile and compliant 3-DOF architecture investigated here appears to be SPH@CG since it performed well in all four maneuvers. In addition the HPR@PH and SPH+HF performed well enough to deserve further investigation.

3. The weak point of the 3-DOF simulators investigated here is, as could have been foreseen, their inherent incapability to simulate high frequency translational accelerations. Although the pilot can be subjected to high frequency specific forces if the vector $\vec{R}0$ is non-zero, these motions are coupled to high frequency angular accelerations. Thus, they cannot be controlled independently, in particular when using a conventional washout filter algorithm.

4. Performing the washout at the pilot's head removes the spurious cues generated by washout artifices such as tilt coordination. However, it also obviates the possibility of using the HF emulator to simulate high frequency translational accelerations in the 3-DOF motion bases.

Future work will entail test sessions with airline pilots on a real simulator (whose motion can be constrained in order to simulate a 3-DOF system), as well as the design of a 3-DOF simulator. It would also be of interest to develop a completely new washout algorithm, which could be optimized to achieve the best performance from a 3-DOF motion base.

6 Acknowledgments

The work reported here was funded by the Natural Sciences and Engineering Research Council of Canada (NSERC) under a Collaborative Projects Grant.

References

[1] Crassous de Medeuil, C., *"Évolution des simulateurs d'avions civils."*, Onde Electrique, VOL 68, No 6, Nov-Dec 1988, pp 35-41.

[2] Lapiska, C., Ross, L. and Smart, D., *"Flight simulation. An overview"*, Aerospace America, Vol. 31, No 8, August 1993. pp 14-17, 33.

[3] Shiabev, V.M., *"New concept of the motion system for the low cost flight simulator: Development and design."*, presented at the 1993 AIAA FST Conference, 1993.

[4] Yang, P.H, Waldron, K. J. and Orin, D. E., *"Kinematics of a Three Degree-of-Freedom motion Platform for a Low-Cost Driving Simulator"*, Proceedings of the 5th Symp. on Advances in Robot Kinematics, Portorož, Slovenia, June 1996.

[5] Repperger, D. W., *"Study of supermaneuverable flight trajectories through motion field simulation of a centrifuge simulator."*, Journal of Dynamic Systems, Measurement and Control, Transactions of the ASME, Vol 114, No 2, June 1992, pp 270-277.

[6] Leung, Y. M., *"Solution of the General Flight Equations in Real Time"*, Master Thesis, Department of Aerospace and Engineering, University of Toronto, 1985.

[7] Hanke, C. R. and Nordwall, D. R., *"The Simulation of a Jumbo Jet Transport Aircraft, Volume 11: Modeling Data"*, NASA CR-114494, Sept 1970.

[8] Gum, D. R., *"Modeling of the Human Force and Motion-Sensing Mechanisms"*, AFHRL-TR-72-54, June 1973.

[9] Nahon, M. A., Reid, L.D., Kirdeikis, *"Adaptive Simulator Motion Software with Supervisory Control"*, Journal of Guidance, Control, and Dynamics, Vol. 15, No 2, March-April 1992, pp 376-383.

[10] Reid, L. D. and Nahon, M. A., *"Flight Simulation Motion-Base Drive Algorithme: Part 1 - Developing and Testing the Equations"*, UTIAS no. 296, chapter 3, December 1985.

[11] Reid, L. D. and Nahon, M. A., *"Response of Airline Pilots to Variation in Flight Simulator Motion Algorithm"*, Journal of Aircraft, Vol 25, No 7, July 1988, pp 639-646.

AIAA-96-3375-CP

Determination of the Physical Volume of a Moving-Base Simulator

S.K. Advani[1], J. E. Mebius[2]
International Centre for Research in
Simulation, Motion and Navigation Technologies SIMONA
Kluyverweg, 2629 HS Delft, The Netherlands
tel +31 15 278 1395 fax +31 15 278 6480 E-Mail: s.advani@ lr.tudelft.nl

Abstract

This paper describes two approaches to describing the physical volume required for the operation of a motion-base simulator. This volume can have a significant impact on the simulator facility requirements (building size). The techniques can also be applied to determine and prevent mechanical interferences between the motion system and other components on the simulator. First, the direct approach, using the inverse-actuator transformation technique, is applied. This leads to a large total number of points, however leaves uncertainty in the exact limits of the motion system volume. Then, an optimization-oriented approach will be discussed, so that only the geometric parameters of relevance to the problem are taken into consideration. This appears to provide a more promising solution. A design example, the SIMONA Research Simulator, is shown, where considerations to the physical volume were critical, also due to the low positioning of the visual display system with respect to the motion platform.

Introduction

The physical volume required by a moving-base simulator, including all of its payload, can be described by the set-theoretic union of the spatial extents of the simulator in all possible locations. The difficulty is in the determination of *all* possible locations, since this is a six-dimensional abstract space of a highly non-linear nature (configuration space). Furthermore, this space has a five-dimensional boundary which has a complicated structure due to the minimum and maximum lengths of six actuators.

Problem Formulation

Whenever the simulator is indeed at an extreme location (the simulator touches the surface of the physical volume somewhere), at least one of its actuators is fully extended. Therefore, to delineate the physical volume one needs at most to travel the boundary of the configuration space.

Without much loss of generality, one may confine oneself to convex cylindrical symmetric superstructures (i. e. symmetric about the perpendicular to the moving platform at its centre). Then the problem possesses the symmetry of an equilateral triangle, and one needs to explore only one-sixth of the boundary space. This is still complicated enough.

A very important simulator building cost factor is the height of the ceiling: ceiling height in excess of the maximal simulator height is useful for allowing maintenance cranes to pass overhead, however this space should be limited as much as possible. In the case of the SIMONA Research Simulator, an experimental facility being developed by the Delft University of Technology, there were constraints in all directions. The visual display system on this device makes use of a ten-foot radius spherical section (see Figure 1 and 2). While traditional

[1] Assistant Professor / Member AIAA
[2] Associate Professor of Mathematics

simulator design makes use of layered structures (Figure 3), their vertical clearance requirements (with the building ceiling) pose the main threat. In the SRS configuration, however, emphasis is placed on the motion bandwidth, and the amount of on-board equipment is restricted. The design of this simulator takes into account the negative effects of parasitic motions by lowering the pilot eye reference plane, and by applying a low-mass high-rigidity composite construction to the main structure. As a result, the display system comes close to contact with the ground, when the simulator is in its maximum nose-down pitch condition. In addition to this, budgetary considerations limited the depth of the new SIMONA laboratory to 12.0 metres. Increasing this dimension would have required the use of non-standard and costly concrete floor structures. Considering the size of the display system (6.1 m diameter), and the maximum displacements of the motion system (e.g. ± 1.1m sway; + 1.3 m surge, -1.0 m surge), attention to the exact volume in *all directions* became critical.

Solutions

A The Inverse Transformation Approach

The direct approach to solving the problem involves mathematically describing the kinematics of the motion system[1]. An inverse actuator transformation can be applied, in order to describe the platform centroid position and Euler angles (with respect to the fixed-base reference frame). This iterative process can be solved using the Newton-Raphson technique[2]. Then, the actuators can be divided into discrete segments. The end of each segment represents a point along the operating stroke of the actuator. An iteration cycle can then be applied to determine the platform position and rotations for each position of the actuator. Vector mathematics can be applied to locate specific critical points of the platform geometry. Normally, these points will include the display system, projectors, and the extreme corners of the simulator platform.

Figure 4 shows the solution of the aforementioned technique when the actuator is sub-divided into nine segments (thus, ten points along the operational length), and twelve extreme points are given on the simulator. Clearly, most of the data is useless in knowing the extremities. The critical areas, however, are rather roughly described. Spaces between the points along the outer boundary are large. One could even argue that the extremes are ill-defined.

B Mixed Forward-Inverse Transformation Solution

While this technique can be applied to the entire volume, the discussion which follows pertains to the maximal height of the simulator alone. In this case, one has to solve an optimization problem to determine the maximum value of the height of vertices of the superstructure as a function of the six actuator lengths with the constraint that at least one actuator (which may be fixed beforehand due to the symmetry of the problem) is maximally extended.

The following solution is proposed:

(1) Make an educated initial guess of the configuration of its maximum height: to this end, choose the vertex furthest from the platform axis, and not too high above the platform, and make tilt the platform, bringing this vertex to near its maximum height;

(2) Extend one of the free actuators until the vertex does *not* rise anymore;

(3) Repeat this for the other free actuators, leaving the previously extended actuators at their current lengths;

(4) If another vertex or vertices rose above the current vertex during these operations, then change to the highest of them;

(5) Repeat the cycle of the preceding three items until the superstructure does not rise anymore.

This procedure lends itself well to a semi-automated computer-assisted execution.

Virtual mock-up

It was decided to design a semi-automated computer-based tool to replace a physical scale-model of the flight simulator. One should be able to specify(a) the geometrical parameters of the simulator motor, and (b) the geometry of the superstructure. Like a real scale-model, the tool should function like the real simulator, i. e. it should show in some way what happens when the simulator moves through its full range of motion allowed by its actuators. This virtual mock-up was constructed under MS-DOS and the ARTIBODIES system for multibody kinematics. ARTIBODIES is a system for the creation, visualization and manipulation of articulated bodies[3]. The SRS served as an example. This led to the name ABSRS, which means SRS based on ARTIBODIES.

Within ABSRS the simulator is implemented as a single link whose vertices are the connections of the lower and upper gimbals, and whose edges are the six actuators and the sides of the lower and upper triangles. Thus, the gimbal pairs coincide in this case. Normally ARTIBODIES links represent rigid bodies; the flexible character of the simulator is implemented by letting the body-based coordinates of the upper gimbal connections vary as functions of the location of the motion platform with respect to some neutral location.

The geometrical parameters of the motor are set to some initial value; they may be modified at any time via an interaction panel. The optional superstructure is specified in an input file to ABSRS; it is represented as an ARTIBODIES tree (which may be a single link) rooted onto the simulator link using the orientation of the motion platform and the position of its centre; in this way the superstructure is fixed to the motion platform.

In ABSRS, the simulator may be moved in two ways:

(a) by changing position and orientation of the motion platform; the actuator lengths required are calculated directly and shown in the interaction panel;

(b) by changing the actuator lengths: a real simulator will by nature assume its new position and orientation, a virtual simulator lacks the physics of a real one, and one needs to calculate the position and orientation from the actuator lengths. This involves six non-linear equations in six unknowns.

In ABSRS these equations are linearized by means of a Jacobian consisting of difference quotients. The position and orientation increments are solved from the actuator length increments by applying the inverse Jacobian; this yields an approximation to the true position and orientation coordinates. No iterative scheme has yet been implemented. This approximation is used to obtain new values of the actuator lengths. These differences are of the order of (increments in cm) x 0.2% of the increments. This figure is deemed good enough for a first exploration of the possible use of the virtual mock-up.

One or more iterations will presumably improve this situation, however one does have a consistent set of position and orientation coordinates and actuator lengths; so the fidelity of the representation of the real simulator by the virtual mock-up is not misleading.

Experimental results

An ABSRS session was conducted with the geometrical parameters and the superstructure of the actual SIMONA Research Simulator. With an eye on the computer picture, it was intuitively clear that the initial guess for the maximal height must come from a nearly maximal roll angle of the motion platform. Within about 20 minutes of working on the actuator lengths, a figure for the maximal height was obtained that apparently could not be improved - at least not within the vicinity of the current set of actuator lengths.

It is noteworthy that not only the maximal actuator lengths but also the minimal lengths come into play; practically this means that only when one can be absolutely sure that the actuators cannot be pushed through their lower buffer zones, one can safely apply a correspondingly lower figure for the height of the simulator housing.

In the ABSRS version used for that session the gimbal offsets are zero; therefore, the results obtained cannot yet be claimed as accurate. Furthermore, no exhaustive search of all combinations of actuator lengths was yet performed. However, there is a good reason to believe that the figure obtained indeed is the maximal height: several completely different initial guesses (nearly maximal pitch; both pitch and height of platform centre about half-way their maximal values etc.) yielded maximal-height figures far below the figure obtained from nearly maximal roll.

Numerical results

With a lower radius of 155.00 cm, an upper radius of 160.00 cm, actuator lengths ranging from 208.80 cm to 323.80 cm, a neutral height of the motion platform of 237.00 cm, and the point (-23.690, -314.450, -238.800) (in cm relative to the usual aeronautical Cartesian coordinate system fixed to the motion platform at its centre) as an extreme point, one obtains a presumably maximal height of 593.465 cm, which is reached for the following position in cm and orientation (aeronautical Euler angles) in degrees:

x = 11.945, y = 32.637, z = 11.813, psi = 9.725, theta = 6.350, phi = 33.716, yielding h-max = 593.465

Conclusions

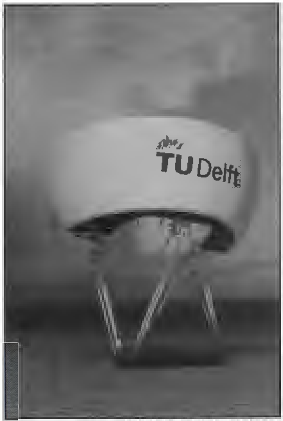
Determination of the physical volume of a simulator is often necessary for the design of the system, or for eventually being able to install the simulator in a given facility. Considering that the cost of simulator facilities is a significant contributor to the hourly operating cost of the training device, efforts to minimize the volume of the building could be of value to the training community. This paper has reviewed two techniques for the determination of this form, and should provide the reader with possible solution approaches.

References

[1] Advani, S.K., and Djavdan, P. "Sensitivities in Simulator Motion Base Kinematics Design". AIAA-96-3477-CP. From AIAA Flight Simulation Technologies Conference, San Diego, July, 1993.

[2] Dieudonne, J.E., Parrish, R.V., Bardusch, R.E., "An actuator extension transformation for a motion simulator and an inverse transformation applying Newton-Raphson's Method". NASA Technical Note TN D-7067, 1972.

[3] J.E. Mebius: Applications of quaternions to dynamical simulation, computer graphics and biomechanics. Ph.D. Thesis, Delft University of Technology, Delft, 1994.



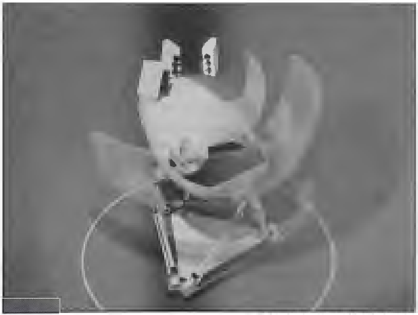


Figure 1. SIMONA Research Simulator (artist's impression). Note that the relatively low placement of the visual display system structure poses considerable concern regarding contat with the ground.

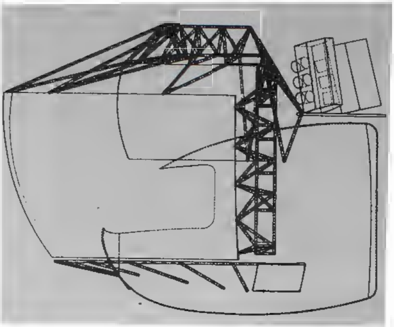


Figure 2. Elevation-view of SIMONA Research Simulator, showing low placement of visual display system, and high upper structure

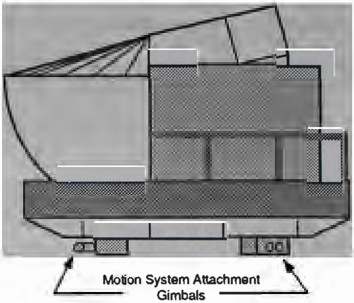
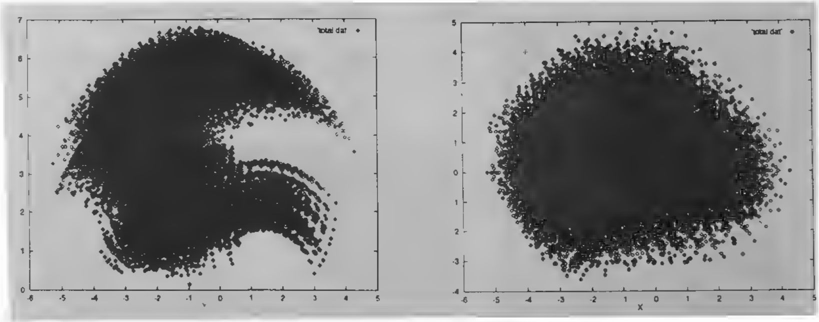


Figure 3. Elevation-view of a typical training simulator (motion system not shown)



(a) Elevation view (from left side)

(b) Plan view (from above)

Figure 4. Collection of points describing motion envelope of SIMONA Research Simulator, calculated using inverse-actuator transformation technique. Actuators divided into ten segments. Twelve critical points on simulator shown.

AIAA-96-3476-CP

A ROBUST CONTROLLER FOR A DYNAMIC SIX DEGREE OF FREEDOM FLIGHT SIMULATOR

Moshe Idan* David Sahar†
Department of Aerospace Engineering
Technion, Israel Institute of Technology, Haifa, Israel

Design and performance of a robust controller for a six degrees of freedom moving base flight simulator is presented. The control objective is to provide the pilot with realistic flight motion cues in the simulator cabin. The robustness requirements result from the uncertainties and variations in the simulator dynamics caused mainly by the uncertainty in the subject's weight and changes in the simulator inertial properties due to changes in the equipment installed on the simulator. The controller design is based on a linearized model of the simulator dynamics together with an uncertainty model that describes the variations of the simulator weight. The main performance specification is for the pilots sensed accelerations and angular rates to track closely simulated aircraft responses, while keeping the simulator motions and controls within allowable limits. The tracking accuracy requirement is frequency weighted and is based on the frequency response characteristics of the human's vestibular system. The design was performed using the μ synthesis and analysis techniques. The performance of the resulting controller is evaluated using extensive nonlinear numerical simulation results.

I. Introduction

MOTION-BASE flight simulation is an essential technology in pilot training and research in the area of man-machine systems. In addition to the visual and instrument cues, a high fidelity flight simulator must provide the pilot with realistic motion cues corresponding to actual flight, while remaining within the simulator's motion limits. To reach this objective, an algorithm , commonly referred as the "wash-out" filter, is needed to transfer the simulated aircraft motions and the resulting motion cues into achievable simulator motion commands. The design of the washout algorithms is complex especially for synergistic six-degrees-of-freedom (6DOF) motion base flight simulators due to the inherent coupling of the required rotational and translational motions.

Many control synthesis techniques were applied for the washout and overall simulator motion control system design, ranging from the classical,[1,2] to adaptive,[3,4] combination of the two,[5] linear optimal control,[6,7] quasi-optimum design[8] and many more. A comparison of these techniques can be found, among others, in Ref. 9. There it is stated that most designs can be tuned to provide satisfactory performance and the difference is in the ease to obtain the desired simulation fidelity.

An additional difficulty in the washout filter design could result from uncertainties in the simulator dynamics. The uncertainties are due to variations in the pilots' weight, changes in the simulator inertial properties due to changes in its setup and installed equipment, uncertainties in the drive system dynamics and more. These uncertainties could turn out to be critical especially in limited power research simulator, such as the 6DOF motion base flight simulator of the Philadelphia Flight Control Laboratory at the Department of Aerospace Engineering, Technion - IIT.

In this paper a robust control design strategy for the motion base of a synergistic 6DOF flight simulator is presented. The controller design technique relies strongly on the simulator's dynamics and uncertainty models to provide robust performance which accounts for the human's sensory system characteristics. Accurate modeling and performance requirement specifications enable favorable performance versus robustness compromises in the controller design.

The controller design strategy is introduced in the next section. In section III the simulator's dynamics model is presented, followed by the actual controller design setup for the Technion 6DOF motion base simulator. The resulting lateral motion controller and its performance are presented in section V.

*Senior Lecturer, Member AIAA

†Graduate Student

II. Controller Design Concept

Design of a controller for the motion base of a high fidelity flight simulator presents a unique and difficult challenge, caused by conflicts in the design requirements. The main goal of the controller is to provide the pilot with realistic wide-band motion cues to better simulate the actual flight environment. This can be only partially achieved due to the simulator's motion limitations and physical constraints, imposing limitations mainly on its low frequency performance. In addition, high frequency performance are determined by the power limitations of the motion drive system. The controller complexity increases even more for the today commonly used synergistic 6DOF motion base flight simulators with strong coupling and stringent synchronization requirements of the different actuators (usually six or more), caused by strongly coupled translational and rotational motions of the simulator.

The design difficulty can be somewhat reduced if it accounts for the characteristics of the human's motion sensory system, mainly the vestibular system. This way the motion performance requirements can be limited only to the relevant frequency range. In addition, the model of a particular aircraft dynamics, which dictates the characteristics of the pilot sensed motion cues, can aid in defining the controller design specifications and reduce possible overdesign. Thus, the controller design strategy suggested in this work will be based on the simulator dynamics, human vestibular system and reference aircraft dynamics models.

Using these models, the controller designer can often be faced with unavoidable model uncertainties which can lead to robust controller design requirements. This problem becomes pronounce when dealing with a limited power research flight simulator, as is the case with the 6DOF motion base flight simulator of the Philadelphia Flight Control Laboratory at the Department of Aerospace Engineering, Technion - IIT. The simulator motion base has a hexapod setup driven by twelve electrical DC torque motors. The motion is limited to about ±15-20 degrees and about ±0.3 meters in rotational and translational degrees of freedom, respectively. The simulator is used mainly for research in the areas of man-machine systems, display technologies and more.

The electrical drive system was chosen because of its relatively simple maintainability and low cost, particularly suitable for a university research environment. The main drawback of the system is its limited power, which led to a light weight cabin design. Consequently, possible variations in the cabin weight due to uncertainty in the subjects weight or changes in the setup of the cabin equipment (adding displays, sensors, etc.) must be carefully accounted for in the simulator motion controller design. In addition, uncertainties in the model parameters and the complexity of the electrical drive system may introduce uncertainties in the overall model of the simulator dynamics.

These system and model uncertainties motivated the robust controller design strategy presented in this work. As will be shown in the sequel, the structured nature of simulator model uncertainty block led to the use of the μ synthesis and analysis techniques. The robust performance requirements are based on the model of the simulator dynamics and its limitations, a reference model of the aircraft response and the human vestibular system model, discussed next.

III. Simulator Dynamics Model

The motion base of the simulator treated in this work is a Stewart platform with an hexapod setup. The length of each leg of the hexapod is controlled by two electrical DC torque motors in the range of 1 to 1.5 meters, approximately, generating limited 6DOF motion with a bandwidth of about 8Hz and a maximum accelerations of nearly 0.5g. The power supply circuitry to the motors consists of a current and voltage loops, in addition to position and velocity loops obtained from potentiometer and tachometer measurements, respectively. Thus, each DC motor operates in a servo loop that controls the length of the hexapod leg and the input voltage is proportional to the required length of the corresponding leg. For each leg, both servo loops are commanded with the same voltage, thus effectively the simulator's motion base has six actuators and six control inputs.

The motion controller will be designed for a linearized model of the simulator dynamics, which accounts for the above mentioned uncertainties. The linearization of the model is around a nominal level mid-stroke orientation of the simulator. The strong nonlinearity of the simulator dynamics model results from the nonlinearities of the kinematics, the electrical drive system and the geometry of the hexapod system. Thus, first the nonlinear model of the simulator dynamics is discussed, followed by the linear model that includes also the inertial uncertainties of the system.

Nonlinear Dynamics Model

A full nonlinear model of the 6DOF motion base flight simulator with an electrical drive system was presented in Ref. 10. This model includes the full 6DOF nonlinear kinematics, the hexapod geometry and a model of the DC servo drive system. The inputs to the model are the six commands to the DC servo loops. It was shown[10] that an approximate second order dynamics model of these servo loops can be included in the overall model using only the rotational and translational degrees of freedom of the simulator, without the need to introduce additional differential equations or degrees of freedom. Thus, a total of six degrees of freedom were used in this model, which were transformed into twelve first order nonlinear state equations with six inputs.

The outputs of the model are divided into two groups: the outputs of actual sensors installed on the simulator and used as inputs to the controller to be designed, and a set of outputs that will define the simulator performance requirements and limitations. The simulator is equipped with a set of inertial sensors: three translational accelerometers and three rate gyroscopes mounted on the simulator's floor. In addition, a potentiometer is installed on each leg of the hexapod to measure its length. Due to the lack of inertial attitude and position sensors, these six length measurements were used to determine the simulator's attitude and position. There is no known analytical solution to this "direct" kinematics problem for a general Stewart platform, except for very special cases and configurations.[11] Even then, a solution of high order polynomial equation is required, accompanied with many non physical solutions which have to be sorted out. In addition, these cases do not match the actual setup of the simulator and therefore in this work the kinematics problem is solved numerically.

The algorithm is based on the inverse model of the hexapod kinematics, which determine the lengths of its legs as a function of its attitude and position. The latter are computed by a numerical search using Newton-Raphson procedure with cubic line search until the computed leg lengths match the measured quantities. In the real-time implementation this search is performed at the controller's update rate of 100Hz. Since the simulator dynamics is much slower relative to this update rate, its attitude and position change very little between two sequential computations and thus the numerical search algorithm, which uses the previous search results as its initial guess, converges rapidly and is suitable for real-time implementation (see Section VI for details). Thus, the hexapod geometry measurements together with this numerical algorithm provide the required attitude and position measurements.

All together, the model has 12 measured output variables (three translational accelerations, three angular rates, three attitude angles and three position coordinates) that will be passed into the control algorithm. The simulator performance specifications are defined by the specific forces and angular rates and accelerations at the estimated pilot head location that model the motion cues that would be sense by the pilot. This adds additional 9 output variables. The simulator limitations will be defined based on the hexapod geometry and attitude and position limitations and thus no additional outputs are required. Most of the above mentioned output variable involve nonlinear computations due to the sensors' geometry, however no additional states and differential equations are required to express these measured/sensed variables.

The accuracy of the nonlinear model was checked by applying the same voltage inputs to the simulation program and the actual simulator in open loop. The input voltage commands to the servo circuits were computed using the inverse model of the simulator kinematics, supplied with the desired representative 6DOF motions: six segments, four seconds each, with almost pure oscillatory motions in roll, pitch, yaw, surge, sway and heave. The numerical simulation fidelity was checked by comparing the computed translational accelerations and angular rates with the actual measurements. Results of these evaluations are presented in Figure 1. The extremely good match between the experimental and numerical data clearly validates this nonlinear model and its parameters.

This nonlinear dynamic model was used to simulate the simulator response which aided to evaluate the motion base control system off-line, before its actual implementation on the actual system.

Linear Model

Due to the inherent complexity of the nonlinear dynamics model of the simulator, its analytical linearization was not possible even when using symbolic programs such as *Mathematica*^{TM}. Thus, the linear model of the 6DOF flight simulator dynamics was obtained by numerical linearization (differentiation) of the full nonlinear model. The resulting state space model has 12 states, 6 inputs (voltage commands), the 21 outputs described above and is

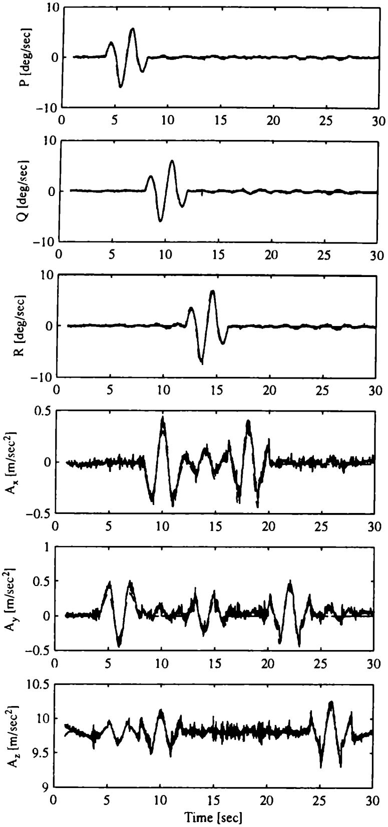


Figure 1: Validation of the nonlinear model of the simulator's dynamics: —— measured data and — — — numerical simulation results.

given by

$$\left\{ \begin{array}{c} \underline{\dot{x}} \\ \underline{y} \end{array} \right\} = \left[\begin{array}{cc} A & B \\ C & D \end{array} \right] \left\{ \begin{array}{c} \underline{x} \\ \underline{u} \end{array} \right\} \triangleq \bar{\mathcal{P}} \left\{ \begin{array}{c} \underline{x} \\ \underline{u} \end{array} \right\} \tag{1}$$

For simplicity, in this work it was assumed that simulator model uncertainties result only from variations in the subject's weight, δ_w. It should be realized though that similar modeling techniques can be applied if addition inertial properties of the simulator are uncertain or are expected to vary during the simulator's extended use. The uncertainty model of the system was obtained by linearizing the nonlinear model while varying the subject's weight, resulting in variations/uncertainty range for each parameter of the state space model (the elements of the $\bar{\mathcal{P}}$ matrix in Eq. (1)) as a function of that weight. It was found that a total of 180(!) parameters are affected by the variation δ_w in the weight. Since all the uncertainties depend on this one unknown physical quantity, the dimension of the repeated scalar diagonal uncertainty block was reduced to minimum using the technique presented in Ref. 12. It drastically reduced the dimension of the uncertainty block from 180 to 17 without affecting the modeling accuracy. This can be expressed in a compact form using a linear fractional transformation (LFT) representation[13] as

$$\mathcal{P} = \mathcal{F}_l(M, \delta_w I_{17}) \tag{2}$$

where I_{17} is a 17 × 17 identity matrix. M is defined by

$$M \triangleq \left[\begin{array}{c:c} \bar{\mathcal{P}} & M_{12} \\ \hdashline M_{21} & M_{22} \end{array} \right] \tag{3}$$

where the $M_{i,j}$ matrices express the dependence of $\bar{\mathcal{P}}$ on δ_w. Note that, as expected, $\mathcal{P} = \bar{\mathcal{P}}$ for the nominal weight, i.e. $\delta_w = 0$.

This linear model is next used for the design of the motion-base controller of the 6DOF flight simulator.

IV. Controller Design Setup

The control design objective is to obtain a constant gain linear controller of the simulator's motion base so that the motion cues sensed by the subject are close to those of a computer simulated aircraft response. This leads to an explicit model following design scenario. Since the simulator model includes uncertainties, a robust controller is required. Due to the structured nature of the uncertainty model (repeated scalar block in Eq. (2)), μ synthesis and analysis techniques were used for the design.[13,14]

The interconnection model used for the controller design is presented in Figure 2. It incorporates the

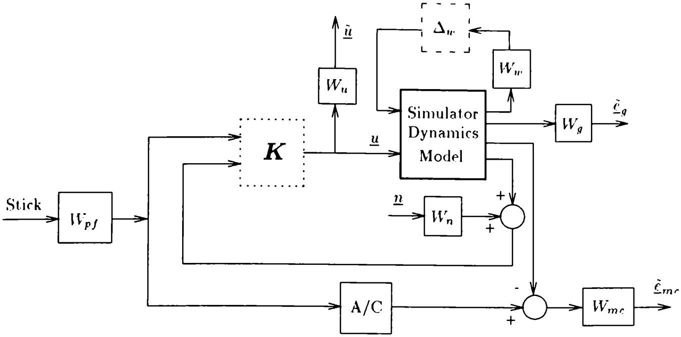


Figure 2: Simulator motion controller synthesis interconnection model.

linear model of the simulator dynamics with its uncertainties, a typical fighter aircraft model and various frequency weighting performance functions. To demonstrate the potential of the proposed design strategy, initially only lateral motion cues caused by aircraft lateral motions were controlled. A full 6DOF motion controller is the subject for further research.

The simulator dynamics model includes uncertainties, modeled by a repeated scalar block of dimension 17 given in Eq. (2). To scale the uncertainty block so that $\|\Delta_w\| \leq 1$, the weighting parameter W_w was introduced. In this work it was assumed that the subjects' weight can vary in the range of 60 to 100kg. Thus the nominal linear system was calculated for a subject's weight of 80kg and W_w was set to $20\, I_{17}$, leading to $\Delta_w = \delta\, I_{17}$ and $|\delta| \leq 1$.

The electrical servo system can be commanded in the range of ±3.7Volts. To ensure controller commands in this range, the weight W_u was set to $1/3.7\, I_6$. The controller is required to provide $|\tilde{u}_i| \leq 1\ \forall\ i = 1, 2, \ldots, 6$, which ensures that the control inputs $\underline{u}$ are within the specified working limits. This prevents the possibility of undesirable controller saturation, which can cause dangerous limit cycles and even instability.

The geometric limitations of the simulator are included in the design setup through the weighting function W_g. It consists of two parts: limits on the extent of the hexapod legs around their nominal working point (the middle point of their full travel range), and limits on the simulator's translational and rotational motions. For the Technion 6DOF motion base flight simulator the limits are: ±0.2m travel of each hexapod leg around its mid point; ±(0.30,0.33,0.23)m of surge, sway and heave translatory motions, respectively; and ±(15,14,22) degrees of roll, pitch and yaw rotational motions, respectively. Thus, the weighting function W_g was constructed as follows

$$W_{legs} = 1/0.2\ I_6 \tag{4a}$$

$$W_{tran} = \mathrm{diag}\{0.30, 0.33, 0.23\}^{-1} \tag{4b}$$

$$W_{rot} = \mathrm{diag}\{15, 14, 22\}^{-1} \tag{4c}$$

$$W_g = \mathrm{diag}\{W_{legs}, W_{tran}, W_{rot}\} \tag{4d}$$

As in the case of the control inputs, since the design requirement is $|\tilde{e}_{g,i}| \leq 1\ \forall\ i = 1, 2, \ldots, 12$, the geometrical limitations will be obeyed.

The main performance objective of the controller is to simulate (follow) the motion cues of a reference aircraft model, the A/C block in Fig. 2. In this work, the linear lateral model of the DC-8 aircraft in a hold pattern (altitude of 15000ft Mach number of 0.44) was chosen.[15] This model is driven by pilot stick inputs: the elevator and aileron deflection commands. To model the pilot inputs which are naturally limited to about 1Hz, two random zero mean white signals with a spectral density of one are passed through the following frequency weighting function

$$W_{st} = \delta^{max} \frac{s/150 + 1}{s/(2\pi) + 1} \tag{5}$$

$$W_{pf} = W_{st}\ \mathrm{diag}\{\delta_e^{max}, \delta_a^{max}\} \tag{6}$$

where the maximum elevator and aileron deflections are $\delta_e^{max} = 0.5$rad and $\delta_a^{max} = 0.22$rad, respectively.

The performance specifications are defined by the weighting function on the error in the lateral motion cues sensed by the pilot. This weighting function is based on the human's vestibular model frequency response characteristics, thus imposing performance requirements only in the relevant frequency range.[16] In the design process, performance requirements are specified on the angular accelerations and the specific forces at the pilot's head location in the aircraft cockpit and the simulator cabin. Thus, only the frequency domain shape of the vestibular system and not its gain is accounted for in the design process. The models of the otolith and the semi-circular canal systems are modeled approximately by[16]

$$W_{oto} = K_{oto} \frac{s + 0.076}{s + 0.19} \tag{7}$$

$$W_{scc} = K_{scc} \frac{s + 9.2}{s + 0.17} \tag{8}$$

and the performance specification weight function has to have a similar shape to those functions.

Due to the physical limitations of the simulator motion system, the above mentioned performance specifications cannot be met. Consequently, the low and the very hight frequency requirements were reduced and the motion cues' weighting function was set to

$$W_{sf} = \frac{1}{\epsilon_{sf}} \frac{s + 0.005}{(s + 0.3)(s^2/100 + 0.14s + 1)} \tag{9a}$$

$$W_{aa} = \frac{1}{\epsilon_{aa}} \frac{s + 0.15}{s + 4} \tag{9b}$$

$$W_{mc} = \mathrm{diag}\left\{W_{sf}, W_{aa}\right\} \tag{9c}$$

where W_{sf} and W_{aa} and the specific force and angular acceleration weighting functions, respectively, while ϵ_{sf} and ϵ_{aa} are their maximum allowable high frequency errors set to 0.25m/sec^2 and 0.06rad/sec^2, respectively. The high pass term was added due to the simulator drive system bandwidth limitations. Since the design requirement is for each element of $\underline{\epsilon}_{mc}$ to be less than 1, the above weights ensure that the errors in the high frequency range will be within the specified limits. In fact, these weights replace the commonly used wash-out filters.[1,6,9] The difference is that here the motion control system design accounts for this wash-out.

The inputs to the controller $\boldsymbol{K}$, which are the measured variables described in Section III, are assume to be contaminated with additive white noise. The approximate spectral densities of the various sensors are: translational accelerometers - 0.27 mili-g; rate gyroscopes - 0.004rad/sec; attitude and position determination - 4.5×10^{-5}rad 4.5×10^{-5}m, respectively. The latter errors are due to the accuracy of the numerical search algorithm discussed in Section III and were determined numerically. To model these error, $\underline{n}$, twelve random zero mean white signals with a spectral density of one are scaled by the weighting function W_n defined by

$$W_{acc} = 2.7 \times 10^{-4} I_3 \tag{10a}$$

$$W_{gyro} = 0.004\ I_3 \tag{10b}$$

$$W_{att/pos} = 4.5 \times 10^{-5}\ I_6 \tag{10c}$$

$$W_n = \mathrm{diag}\left\{W_{acc}, W_{gyro}, W_{att/pos}\right\} \tag{10d}$$

After including all the performance and robustness specifications, the resulting generalized model consists of 23 states.

V. The Controller Performance

Based on the interconnection model and the weighting functions presented in the previous section, a robust control for the simulator motion base was designed. Deny robust control design technique could be applied here, however due to the structured form of the uncertainty block in the simulator's dynamics model, the design was performed using the μ synthesis technique.[14] The design process converged after three D-K iterations to a $\mu = 0.957$, which ensures robust performance of the controller for all possible model variations. The order of the controller was then reduced from 72 to 36 with only a minor effect on the value of μ.

This controller was tested using the numerical closed loop nonlinear simulation of the simulator's dynamics, presented in Section III. Figure 3 presents the good tracking of the pilot sensed lateral specific force for a maximum magnitude sinusoidal aileron input at 1rad/sec as compared to the simulated aircraft signal. The pilot sensed specific force in noisy due to the measurement noises in the acceleration feedbacks, however the low frequency signal tracks the desired output very closely and thus it can be concluded that the pilot would have been provided with motion cues similar to the reference. In fact, this confirms with the required noise attenuation design requirement. Figure 3 also compares the simulated and reference aircraft roll rates. Since at this initial evaluation stage no performance requirements were specified to this output, the tracking (especially in the steady state) is reasonable. Imposing specification on angular rates and accelerations is the subject of further research and design.

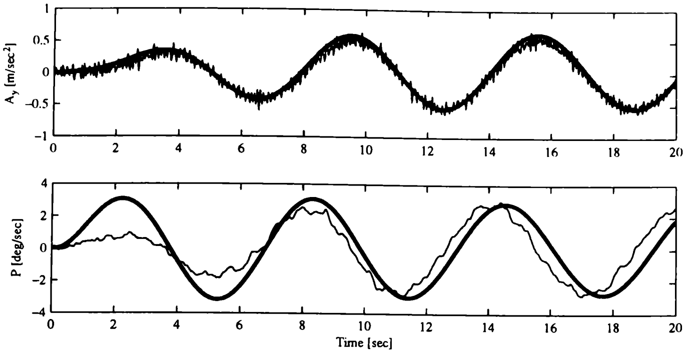


Figure 3: Simulated pilot sensed lateral specific forces and roll rate for a sinusoidal aileron input of 1rad/sec: —— numerical simulation data, ━━ reference model output.

To evaluate the robustness of the design, Figure 4 presents the negligible effect of the pilot mass on the performance. In this figure the pilot sensed lateral specific force is compared for a one second aileron doublet input. The pilot mass is assumed to be maximum, i.e., 100kg. The close tracking after the reference signal (again noisy due to measurement noise in acceleration readings) validates the robust design characteristics.

Experimental Evaluation

The controller presented in the previous section was implemented for tested on the 6DOF motion base flight simulator of the Philadelphia Flight Control Laboratory an the Technion. The evaluation experiments are about to take place in the very near future. In this section the experimental setup is briefly described.

Experimental Setup

The 6DOF motion flight simulator was already described in Section III. The motion controller, which commands the voltage inputs to the six pairs of the DC servo actuators of the simulator, was implemented on a Pentium™ 90MHz PC, equipped with Keithley 12 bit A/D (DAS-1802HC) and D/A (DDA06) boards including a real time clock (RTC).

The inputs to the controller are the simulator cabin accelerations, angular rates, position and attitude (see discussion in Section III.) In the real time implementation, each simulation loop includes: a read and write to the A/D and D/A channels, the position and attitude determination and the controller computations. The latter involves the integration of the linear dynamics equations of the controller using the second order Runge Kutta procedure. Before the actual experiments, the simulation loop was timed using the RTC and was found to be completed in about 5msec. Thus, the simulation rate was safely set to 100Hz to allow for future developments/extensions.

The actual experiments will be performed shortly and reported in subsequent publications.

VI. Summary

In this paper, a robust controller design for a six-degrees-of-freedom flight simulator motions system was presented. The objective is to generate pilot motion cues similar to the actual flight experience. The controller design strategy and performance requirements rely strongly on the dynamics models of its various components of the system. An accurate nonlinear model of the simulators dynamics, tested and evaluated against experimental data, is used as the basis to the design. The model may include uncertainties due to varying characteristics of a research simulator, such as common uncertainties in the simulator inertial properties, thus posing a robust control design problem. The performance specifications are based on models of the human's motion sensory

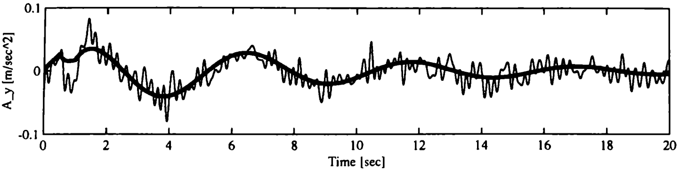


Figure 4: Simulated pilot sensed lateral specific forces for a doublet aileron input for a pilot of maximum mass of 100kg: —— numerical simulation data, ▬▬ reference model output.

system and incorporate simulator geometrical and power limitations. The latter ensures closed loop operation while avoiding actuator saturation. This type of characteristics is mostly desirable for such real time applications. Finally, the performance requirements are adjusted to a particular simulated aircraft in an implicit model following setup.

A robust controller was designed using μ synthesis and analysis techniques. A final value of $\mu < 1$ confirms that the robust performance requirements were obtained. The controller was extensively tested using the nonlinear simulation model, including measurement noise and model variations. It was found that the controller is capable to provide the pilot with motion cues similar to those of the reference aircraft. The controller was implemented on a real time system of the research motion base flight simulator and is currently evaluated in actual motion experiments.

Acknowledgment

The authors would like to thank Dr. Hector Rotstein of the Department of Electrical Engineering at the Technion, IIT, for the invaluable discussions.

References

1. S. F. Schmidt and B. Conard. Motion Drive Signals for Piloted Flight Simulators. Technical Report CR-1601, NASA, May 1970.
2. R. V. Parrish, J. E. Dieudonne, and D. J. J. Martin. Motion Software for a Synergistic Six Degree of Freedom Motion Base. Technical Report TN D-7350, NASA, Dec. 1973.
3. R. V. Parrish, J. E. Dieudonne, R. L. Bowles, and D. J. J. Martin. Coordinated Adaptive Washout for Motion Simulators. *Journal of Aircraft*, 12(1):44–50, Jan. 1975.
4. D. Ariel and R. Sivan. False Cue Reduction in Moving Flight Simulators. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-14(4):665–671, Sept./Oct. 1984.
5. M. A. Nahon, L. D. Reid, and J. Kirdeikis. Adaptive Simulator Motion Software with Supervisory Control. In *Proceedings of the AIAA Flight Simulation Technologies Conference*, pages 97–105, Dayton, Ohio, Sept. 1990. AIAA.
6. W. R. Sturgeon. Controllers for Aircraft Motion Simulators. *Journal of Guidance, Control, and Dynamics*, 4(2):184–191, Mar.-Apr. 1981.
7. R. Sivan, J. Ish-Shalom, and J. K. Huang. An Optimal Control Approach to the Design of Moving Flight Simulators. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-12(6):818–827, Nov./Dec. 1982.
8. B. Friedland, C. K. Ling, and M. F. Hutton. Quasi-Optimum Design of a Six Degree of Freedom Moving Base Simulator Control System. Technical Report CR-2312, NASA, Oct. 1973.
9. M. A. Nahon and L. D. Reid. Simulator Motion Drive Algorithms: A Designer's Perspective. *Journal of Guidance, Control, and Dynamics*, 13(2):356–362, Mar.-Apr. 1990.
10. G. Gandelman. Design of a Control System for a Six-Degree-of-Freedom Dynamic Simulator. Master's thesis, Department of Aerospace Engineering, Technion - IIT, Haifa, Israel, Nov. 1987.
11. P. Nanua, K. J. Waldron, and V. Murthy. Direct Kinematic Solution of a Stewart Platform. *IEEE Transactions on Robotics and Automation*, 6(4):438–444, Aug. 1990.
12. M. Idan and G. E. Shaviv. Robust Control Design Strategy with Parameter Dominated Uncertainty. *Journal of Guidance, Control, and Dynamics*, 19(3):605–611, May-June 1996.
13. J. C. Doyle, A. Packard, and K. Zhou. Review of LFTs, LMIs and μ. In *Proceedings of the IEEE Conference on Decision and Control*, volume 2, pages 1227–1232, Brighton, England, Dec. 1991. IEEE.
14. G. J. Balas, J. C. Doyle, K. Glover, A. Packard, and R. Smith. *μ-Analysis and Synthesis Toolbox User's Guide*. Natick, Mass., 1993.
15. D. McRuer, I. Ashkenas, and D. Graham. *Aircraft Dynamics and Automatic Control*, pages 711–717. Princeton University Press, New Jersey, 1973.
16. R. J. A. W. Hosman and J. C. Van der Vaart. Vestibular Models and Thresholds of Motion Perception: Results of Test in a Flight Simulator. Report LR-265, Department of Aerospace Engineering, Delft University of Technology, Netherlands, Apr. 1978.

AIAA-96-3477-CP

Sensitivities in Simulator Motion Base Kinematic Design

Sunjoo K. Advani[1], ir. P. Djavdan[2]
International Centre for Research in
Simulation, Motion and Navigation Technologies SIMONA
Delft University of Technology
Faculty of Aerospace Engineering
Kluyverweg 1, 2629 HS Delft, The Netherlands
tel. +31 15 78 1395 • fax +31 15 78 6480 • E-mail: s.advani@lr.tudelft.nl

Abstract

The kinematic envelope of a flight simulator motion system is determined by the actuator design, as well as the geometry of the upper (simulator) and lower (earth-fixed) structures. This paper will show the relationship to the resulting limits of the motion envelope as a function of the design variations for a synergistic six-degrees-of freedom motion system. These mathematical functions are highly non-linear, and demonstrate discontinuities, making the effort indeed significant to the effectiveness of the simulator motion system. Small variations in the positioning of the motion actuator attachment gimbals can yield large changes in the resulting mechanical envelopes. Furthermore, the effects on the changes in actuator static and dynamic loads can also be significant. The simulator designer must however always ensure that the configuration meets critical kinematic criteria, such that a given set of actuator lengths yield a unique solution in platform coordinates. The aim of this research is to show how one can make maximum use of the design variables, and ensure that the resulting configuration will operate safely. A design example, the SIMONA Research Simulator of the Delft University of Technology, will illustrate the utilization of these techniques.

Nomenclature

(Boldface indicates vector quantities)

A_0	Moving-platform gimbal centroid
B_0	Base-frame gimbal centroid
$\mathbf{Ar}$	Moving-platform gimbal circle radius
$\mathbf{Br}$	Base-frame gimbal circle radius
D	Half upper gimbal offset
P	Half lower gimbal offset
α	Half upper gimbal subtended angle
β	Half lower gimbal subtended angle
h	Base-frame gimbal angle from B0
k	Moving-platform gimbal angle from A0
$\mathbf{li}$	Vector length of actuator i (from lower to upper attachment)
L_{min}	Length of actuator in minimum operational stroke
L_{max}	Length of actuator in maximum operational stroke
L_{mid}	Actuator mid-length
L_{norm}	Actuator normalized length
T	Euler transformation matrix
$\mathbf{R}$	Vector from A0 to B0
$\mathbf{ri}$	Vector from A0 to upper gimbal attachment

Introduction

Flight simulators which make use of motion cueing devices require the ability of the motion system to provide accelerations to the cab

[1] Director of SIMONA / Assistant Professor / Member AIAA

[2] Research Assistant

which represent those experienced in the flight-vehicle environment. While it is not possible to reproduce these motions with exact magnitude due to the inherent displacement limitations of the motion system, specific forces and angular accelerations can be presented in such a way that the pilot's vestibular system is sufficiently stimulated. The eventual aim is to present these whole-body dynamics, together with the outside-world and instrumentation visual cues such that the control strategies applied by the pilot in the aircraft and in the simulator are similar. This similarity yields a direct transfer of skills, without further need of adaptation.

The underlying requirement of simulator motion cueing is to present the pilot with self-motion cues, meaning relative to the environment, and to ensure that the pilot does not perceive motion of the environment under any circumstances[1]. This problem is complicated by the fact that sensitivities in human motion perception thresholds are influenced by variations in the task and instantaneous workload of the pilot.

While the simulator is limited in its excursions, the temporal fidelity remains crucial. Time delays can yield significant changes in the control strategy and the perception of the handling qualities[2]. The effect is more pronounced when there are temporal differences between the presentation of visual and the vestibular motion cues. While significant technological improvements have been made in the temporal fidelity of motion and visual cueing systems, the ultimate capability of the simulator is confined by the kinematic geometry of the motion system.

The Delft University of Technology (DUT) is developing a research simulation facility called SIMONA[3]. This is a joint venture between the Faculty of Aerospace Engineering, the Faculty of Mechanical & Marine Engineering, and the Faculty of Electrical Engineering. The aim of the full-flight simulator of this facility is to act as a technology development and demonstration platform. In this role, the SIMONA Research Simulator (SRS) will be used to further developments in simulation techniques related to the simulation of flight and ground-based vehicles. In addition to its iron-bird missions, it will be used to study the man-machine aspects of (primarily flight) vehicles. Mathematical models of radio navigation systems will include natural anomalies, allowing researchers to study their effects on pilot behaviour.

The design of this simulator, specified by its multiple applications, requires that the motion cues represent the vehicles and tasks in an acceptable way. While the washout and motion system control software attempt to maximize the system performance for a given set of conditions, the development of the motion system required that its mechanical geometry be optimized. Given the freedom to define the motion system (which has since been designed, fabricated and assembled by the DUT), the author was able to develop a design criteria and later to specify the geometry based on a total set of requirements.

Kinematic design criteria

The kinematic geometry of a synergistic motion platform can be scaled relative to the length and the stroke of the motion system actuators. The minimum and maximum lengths of these actuators, and the lengths of hardware safety buffers within, are a pure function of the actuator mechanical design. Then, maximization of the relative displacements with respect to a given actuator design geometry requires (a) the selection of a criteria by which this maximization can take place, and (b) consideration of the critical conditions of the motion system, both kinematically and dynamically. While the maximization of the displacements is dictated by desired cueing capabilities, failure to meet the critical conditions can lead to complete mechanical failure of the design due to excessively high loads.

Definitions

Figure 1 shows the convention which will be used to define the motion platform geometry. In Figure 2, the vector notation is provided. In the case of the SIMONA Research Simulator, the direction of flight (X-axis) is subtended by A_1 and A_2. Note that this is not always the case with every simulator. The vector relation of this geometry is:

$$\mathbf{l}_i = \mathbf{A}_i + \mathbf{R} + \mathbf{B}_i \qquad \text{[eq. 1]}$$

Note that the length can be given as a normalized vector (Lnorm), rather than an absolute length vector.

The Euler Transformation Matrix is given in Figure 3.

Gimbal Angles

Depending on the type of gimbal design, and namely the degrees of freedom available at the gimbal, the gimbal axis and actuator axis must not co-align to prevent the occurrence of gimbal lock. Co-linearity, or proximity to, can cause severe local loading and excessive strain on the mechanism. A fork-clevis type of gimbal, as commonly used on simulators provides two degrees of freedom.

The mechanism of the motion system must also be configured such that there is no potential for mechanical contact at the gimbals (see Figure 4). This hazard does not present itself exclusively when the actuators are at their maximum or minimum strokes, and it is therefore dangerous to assume when this condition is critical. This is a classic example, where the designer must check all critical conditions, even after slight design alterations.

Generally, the gimbal angle between the actuator central axis and the gimbal central axis can be described by the dot product of (normalized) vectors aligned with these axes.

Since the minimum gimbal angles are a function of mechanical design, specification of acceptable limits are not straightforward and depend on factors like the minimum-achievable bearing diameter (related to the bearing loads), and the gimbal shaft diameter (related to the allowable bending and, hence, stiffness of the shaft). Design of such systems is therefore an iterative and integrated process. SIMONA designed the mechanical upper gimbals to allow a minimum angle of 45 degrees, thereby reversing (and simplifying) the design problem. Subsequent motion system kinematic geometries were checked throughout their envelopes to ensure sufficient mechanical clearance.

The angles τ and γ, which will later be used to define critical kinematic conditions, can be expressed as:

$$\tau_i = \arcsin (T_{31}\, L_{norm}(i,x) + T_{32}\, L_{norm}(i,y) + T_{33}\, L_{norm}\,(i,z)) \qquad \text{[eq. 2]}$$

and

$$\gamma_i = \arcsin(L_{norm}\,(i,z)) \qquad \text{[eq. 3]}$$

Critical conditions

A motion system should never attain (i) a geometric condition which is irreversible, (ii) where the loads are excessive of the mechanical and material properties, or (iii) where there is contact between the mechanical components of the motion system or the moving platform. A given set of actuator lengths should yield only one platform position and orientation. This will be discussed below, classified for various cases, followed by criteria for the static and dynamic loads. Confirming that a design meets the third constraint, mechanical contact, is an extensive subject, and is the topic of another paper[4].

Twist:

This condition defines the point where the platform undergoes a pure or combined yaw rotation such that restoration to the neutral position is impossible. Twist will occur if the upper and lower attachment points of a given actuator pass through the vertical plane. This can be checked by determining the lateral offsets of the actuators when maximum yaw is applied, i.e.,

$$\psi \text{ when } l_1 = l_3 = l_5 = \text{lmin } \textbf{and} \quad l_2 = l_4 = l_6 = \text{lmax} \qquad \text{[eq. 4]}$$

The safety condition necessary to prevent excessive twist is,

$$\left|\psi\right| < 60^\circ + \theta_1 - \theta_2 \qquad \text{[eq. 5]}$$

A safety factor can be applied to the twist condition, limiting the proximity to the absolute minimum allowable angles. Note that θ_1 and θ_2 are mechanical design parameters (see Fig. 1).

Stretch:

If the moving-base gimbal plane is permitted to "fall through" the plane described by the fully-extended actuators as shown in Figure 5, then the stretch condition is reached. The mathematical definition is

$$0 < \tau i < 180° \quad \text{[eq. 6]}$$

Point:

Should the upper gimbals pass through the plane of the lower gimbals as shown in Figure 6, then a Point condition is reached. Although this does not yield a non-deterministic solution to the actuator-platform transformation, contact of the moving-base with the ground must be avoided.

$$0 < \gamma_i < 180° \quad \text{[eq. 7]}$$

Flat:

If the upper platform passes through the plane of the lower platform, then this condition will occur. This is irreversible, and the motion system cannot be restored. Prevention of this condition requires that the ground angles and the moving platform angles are limited between zero and 180 degrees,

$$0 < \tau i < 180° \text{ and } 0 < \gamma_i < 180° \quad \text{[eq. 8]}$$

The maximum likelihood of this can be checked when all actuators are at their minimum lengths.

Actuator Length

The design of the SIMONA Research Simulator (SRS), and perhaps other synergistic devices, is rarely an arbitrary process. The SIMONA actuator design was conceived by the Faculty of Mechanical & Marine Engineering, and employed a double-concentric design (see Figure 7). The stroke was chosen to be 1250 mm maximum, with 50 mm buffers at each end, resulting in a operational stroke of 1150 mm. The mechanical designers were able to package this stroke in a highly-efficient mechanism, yielding a length range from 1338 mm to 2588 mm.

An important consideration in the determination of the actuator length is the trade-off between the length, and the resulting actuator lateral natural frequency. A primary aim of the SRS is to investigate the effects of simulator bandwidth on pilot control behaviour, and to assist aviation authorities in establishing standards for training. This enhancement of current techniques requires a system with capabilities beyond those available today. The bandwidth (and maximum acceleration) of the motion cueing system is limited by the mechanical power available, and the platform mass. The transfer function, and hence the system phase margin, is dependent to a considerable extent on the gross moving load[5]. The load influences the system natural frequency, and can influence the amount of compensation necessary to achieve the required temporal fidelity.

Consideration must be given, in fact, to the entire mechanical system natural frequency: increased rigidity of the structure should not be made redundant by reduced stiffness of the actuator legs. For this reason, the SRS actuators were limited to the values given above. The resulting natural frequency in full extension is estimated at 94 rad/s. This is coherent with the platform natural frequency in a fully-loaded configuration.

Positioning of gimbal attachment points

Given an actuator design, the geometric definition of the upper (moving-base) and lower (fixed-base) mechanisms can be determined. Several design issues are related to (or constrained by) mechanical hardware such as bearing diameters and allowable loads, and all kinematic geometries must be checked for gimbal lock, mechanical interference, and the twist, flat, stretch, point, pinch conditions. The location of the gimbals has a profound effect on the resulting motion limits. Already, we have seen that their selection is not arbitrary, however an understanding of the trade-offs between the degrees of freedom provides insight to the system designer.

The mathematical relationship between the geometry of the gimbal attachments and actuator strokes is complicated by the non-linear relation between actuator lengths and platform position. The transformation from actuator lengths to platform orientation requires solving six simultaneous non-linear equations, best performed by an iterative method[6]. The inverse can be applied to check the actuator lengths, given the platform orientation. Note that this can be a valuable check, if a program is written to move the platform through its envelope and check if there is over or under-extension of any actuator.

Kinematic Design Criteria

While it may be possible to significantly change the maximum displacements in the six degrees-of-freedom, this, like any optimization, implies compromises. In any case, the amount of continuous acceleration will be limited by the displacements available in ground-based simulation: The duration of a sustained linear acceleration in a single degree of freedom is proportional to the square root of the translational acceleration length. While this paper does *not* intend to argue the relative merits of any degree-of-freedom, attempts have been made to understand the importance of motion these. The necessary excursions is a subject of much discussion and research[7]. In any case, since ground-based simulation requires compromises of displacements, attention to the quality of the acceleration cues would appear to be more relevant with the highest yield. In Table 1 summarizes these requirements[8].

Desired excursion properties are limited by the mechanics of the motion system. In the text which follows, it will be clear that the optimization of the motion excursions is, in any case, a limited effort if one compares the relative changes to the displacements encountered in flight. Still, there is considerable benefit to optimizing whatever is available, if only to understand the relative trade-offs.

Motion Excursion Limits

The large number of parameters which determine the eventual limits of a motion system are usually constrained by practical factors. Motion bases are usually symmetric, and often the stroke of the actuator is a given value. The remaining geometric parameters are the upper and lower circle radii (Ar & Br), and the spacing between the upper and lower gimbals (2D and 2P) at each gimbal set. Usually, the upper circle radius, dictated by the platform design, is more critical than that on the ground frame. If an upper circle radius can be established, one has the freedom then to select the other parameters.

Excursion limits can be determined by an iterative method which translates or rotates the platform along a pre-determined path. This path is usually selected parallel to the earth-fixed reference frame. At every step the actuator excursions are checked. Once a limit is exceeded, a bisection method, utilizing a given tolerance, reverses the direction of travel iteratively until a satisfactory convergence is achieved. This is then considered a maxima or minima along that particular line of travel.

The technique discussed above was applied to aid in determining the motion capabilities of the SIMONA Research Simulator. In this particular case, the upper gimbal circle radius had been selected, based on allowing the simulator floor to be "sunk" into the frame of the motion platform. By subsequently lowering the centre-of-gravity of the gross moving load, this design philosophy reduces the compensation required to eliminate crosstalk and phase shifts. The remaining variables were exercised, and the results are summarized below.

The maxima of the excursion limits are described by non-linear relationships, with discontinuities. These discontinuities occur when a particular set of actuators attain their limits, and thereafter become the length-limiting factor as the platform tries to move along a prescribed course. This is particularly true for sway and surge motions. Recall that the actuator length maxima prescribe spheres with the origins at the gimbals.

Figures 8 to 13 show the maximum achievable limits of the motion system. The platform begins in the neutral position (defined when the *actuator* is in its mid-position). For a given lower gimbal spacing P, the lower gimbal circle radius Br is varied. The maximum excursion in a given degree of freedom is then established. This is iterated for changes in Br, and later in P. Note that due to the asymmetry of the motion system through the y-z plane, the maximum and minimum surge and yaw are dissimilar. Heave also exhibits an asymmetric nature.

These figures should imply the flexibility available to the designer in selecting the geometry of the motion system. Minor changes in mechanical design appear to yield major changes in the potential of a given configuration. It is beneficial however to show the inter-relation of these effects (Figure 14). Here, the values Br and P are varied independently. Note that the trade-offs are significant; A larger Br yields significant increases in heave and pitch limits, while sway, surge and yaw drop appreciably. Roll limits also increase, though are not significantly affected. Secondly, an *increase* in P increases the sway, surge and yaw, while the heave, pitch and roll limits are reduced.

Limitations In Design Freedom

While there appears much to be gained by changing the configuration, one must also consider whether a particular design reaches its kinematic limits (twist, stretch, point, flat), and the effective loads on the actuators. Instantaneous dynamic loads can be used then to check the design. It is possible to use vector analyses to determine the effects of acceleration-related loads. Estimation of the true accelerations requires however complex models of the motion system.

SIMONA Research Simulator Motion System

The design approach outlined in this paper has been applied to define the SIMONA Research Simulator Motion System. Considering that this device should sufficiently provide motion cues in configurations representing fixed and rotary-wing aircraft, as well as surace vehicles, made the establishment of the motion system geometry a complicated, and somewhat subjective task. Eventually, since the priority was to simulate flight vehicles, a conventional motion system was selected. The actuator design was dictated by manufacturing capabilities, and the moving platform gimbal circle was optimized with respect to the foreseen cockpit requirements. Note that, in this case, the cockpit is sunk below these attachment points, in order to reduce the effects of inertial coupling. The upper gimbals were placed close together, to minimize the moving load associated with this robust mechanism. Then, by varying the lower gimbal circle radius, and varying the space between these gimbals, the design cycle continued. With each iteration, the critical conditions were checked. The loads were evaluated by performing a vector load analysis.

The final configuration of the SRS motion system is as follows:

Ar = 1.60 m
Br = 1.55 m
D = 0.30 m
P = 0.10 m
Lmin = 2.088 m
Lmax = 3.238 m

An impression of the completed simulator is shown in Figure 15.

Conclusions

The motion system designer has a considerable amount of flexibility in specifying the motion system parameters. The limitations imposed by the critical conditions, and by the allowable loads on the actuators, constrain the amount of freedom.

From this research, one can summarize the conclusions as follows. Note that these are only applicable to the six-degrees-of-freedom Stewart platform geometry:

1. The actuator geometry dictates the motion system geometry, which is scaled relative to the actuators. The use of double-concentric actuators, where the kinematic efficiency (ratio of minimum to maximum length) is high, the kinematic geometry becomes more critical.

2. The envelope in which the kinematics can be varied without encountering the critical conditions, is relatively small.

3. The designer of the motion system must always consider the actuator loads for a given mass distribution prior to finalizing the design. Slight variations to the kinematics can have a significant impact on the actuator and motion system frame loads.

Degree of Freedom		Applications
Surge	X	Road vehicles, VTOL
Sway	Y	Coordinated (turn) washout in aircraft Also represents translational component of yaw.
Heave	Z	Touchdown bump, VTOL handling. Unless excursion is large, substitution by g-seat feasible. Translational component of pitch.
Roll	Φ	Tracking tasks
Pitch	θ	Tracking tasks
Yaw	ψ	Lateral motions due to systems failures, VTOL handling, Dutch roll

Table 1. General motion requirements per degree of freedom (ref. 7 & 8)

[1] Hosman, R.J.A.W. and Steen, H. van der, "False Cue Detection Thresholds in Flight Simulation". AIAA-93-35-CP. From AIAA Flight Simulation Technologies Conference, Monterey, August, 1993.

[2] Mitchell, David G., Hart, Daniel C., "Effects of simulator motion and visual characteristics on rotorcraft handling qualities evaluations". in proceedings of AGARD Conference on Piloted Simulation Effectiveness, , Brussels, October 1991.

[3] Advani, S.K., "The Development of SIMONA: A Simulator Facility for Advanced Research into Simulation Techniques, Motion System Control and Navigation Systems Technologies". AIAA-93-3574-CP. From AIAA Flight Simulation Technologies Conference, Monterey, August, 1993.

[4] Advani, S.K., and Mebius, J.E., "Determination of the Physical Volume of a Moving-Base Simulator". AIAA-96-3475-CP. From AIAA Flight Simulation Technologies Conference, San Diego, July, 1993.

[5] Advani, S.K., and Verbeek, R.J., "The Influence of Platform Inertial Properties on Simulator Motion System Performance". AIAA-94-3418-CP. From AIAA Flight Simulation Technologies Conference, Scottsdale, August, 1994.

[6] Dieudonne, J.E., Parrish, R.V., Bardusch, R.E., "An actuator extension transformation for a motion simulator and an inverse transformation applying Newton-Raphson's Method". NASA Technical Note TN D-7067, 1972.

[7] Rolfe, J.M. and Staples, K.J., "Flight Simulation". Cambridge University Press, Cambridge UK, 1986.

[8] Martin, E.A., "Motion and Force Simulation Systems 1 - Whole-body motion simulators". From AIAA Flight Simulation Update Course Notes. State University of New York, Binghamton, NY, 1995.

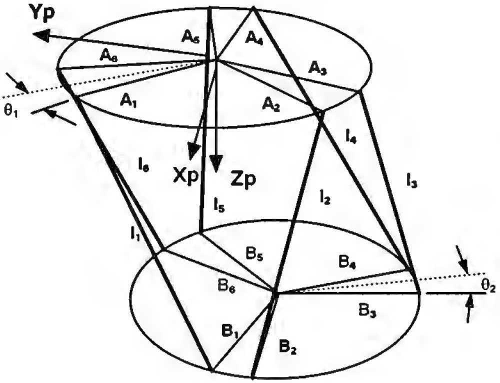


Figure 1. Motion System Labelling and Sign Convention

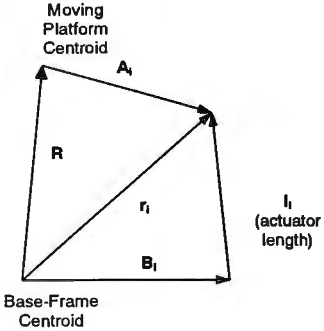


Figure 2. Vectors Defining Actuator Relationships

$$T = \begin{bmatrix} \cos\psi\cos\theta & \sin\psi\cos\theta & -\sin\theta \\ \cos\psi\sin\theta\sin\Phi - \sin\psi\cos\phi & \sin\psi\sin\theta\sin\Phi + \cos\psi\cos\Phi & \cos\theta\sin\Phi \\ \cos\psi\sin\theta\cos\Phi + \sin\psi\sin\Phi & \sin\psi\sin\theta\cos\Phi - \cos\psi\sin\Phi & \cos\theta\cos\Phi \end{bmatrix}$$

Figure 3. Euler Transformation Matrix

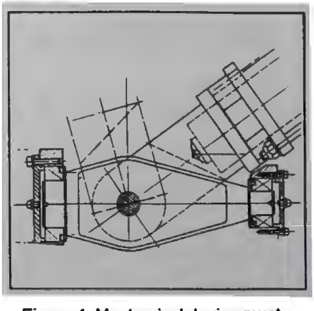


Figure 4. Mechanical design must ensure adequate angular clearance

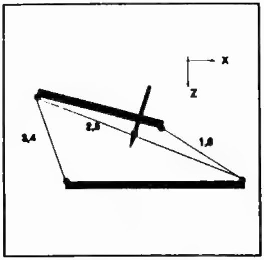


Figure 5. stretch critical condition

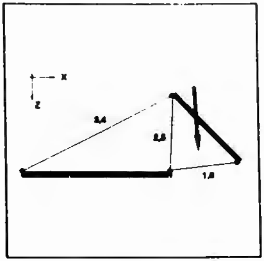


Figure 6. Point critical condition

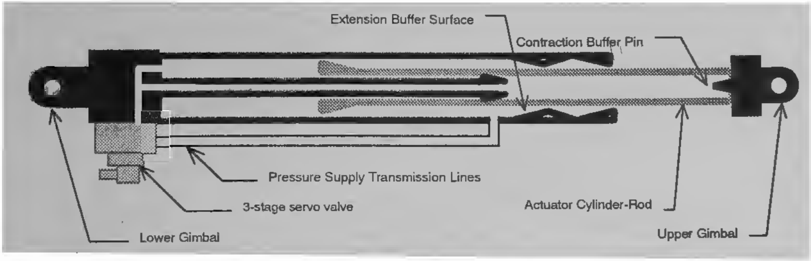


Figure 7. Delft University of Technology linear hydrostatic actuator.
Minimum Length = 2.088 m Maximum Length = 3.328 m

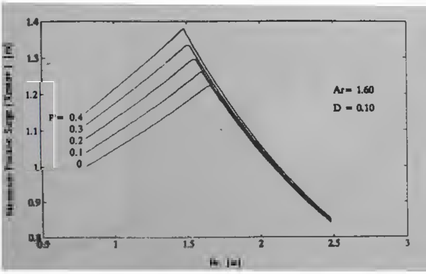


Figure 8(a). Maximum Positive Surge as a function of lower platform parameters

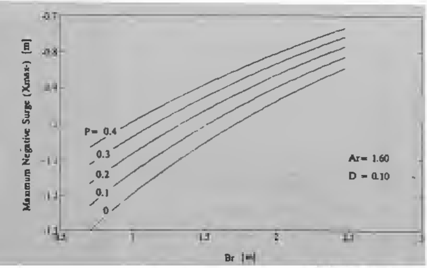


Figure 8(b) Maximum Negative Surge as a function of lower platform parameters

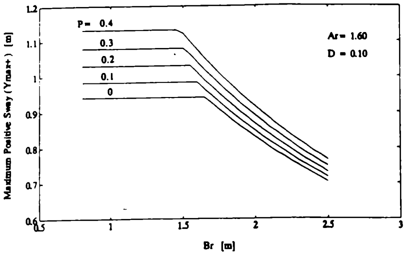


Figure 9. Maximum Sway as a function of lower platform parameters

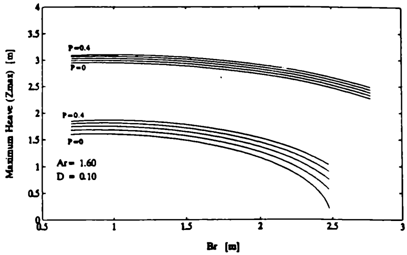


Figure 10. Maximum Heave as a function of lower platform parameters

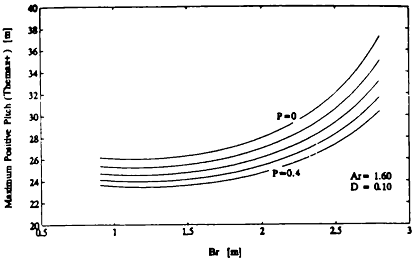


Figure 11(a). Maximum Positive Pitch as a function of lower platform parameters

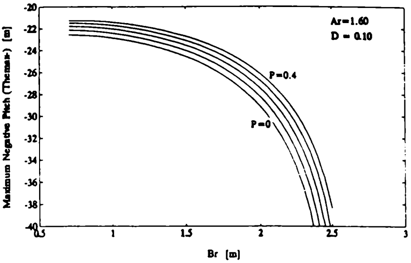


Figure 11(b) Maximum Negative Pitch as a function of lower platform parameters

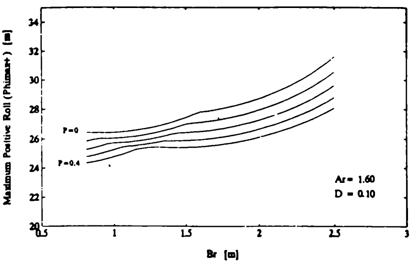


Figure 12. Maximum Roll as a function of lower platform parameters

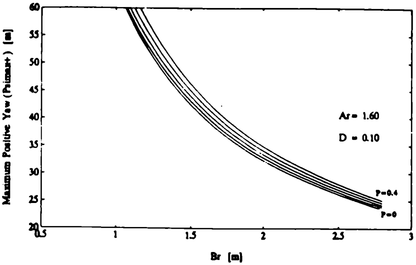


Figure 13. Maximum Yaw as a function of lower platform parameters

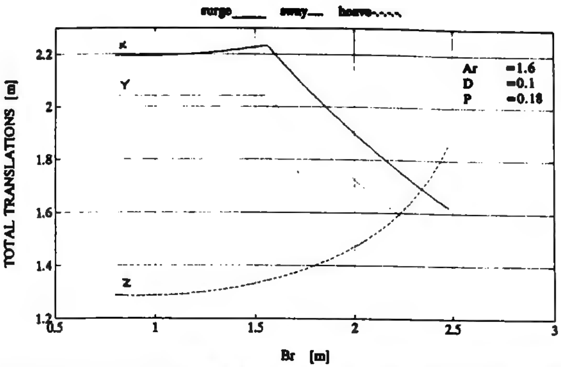


Figure 14(a). Surge, Sway, Heave. Maximum Total Translations as a function of lower gimbal circle radius

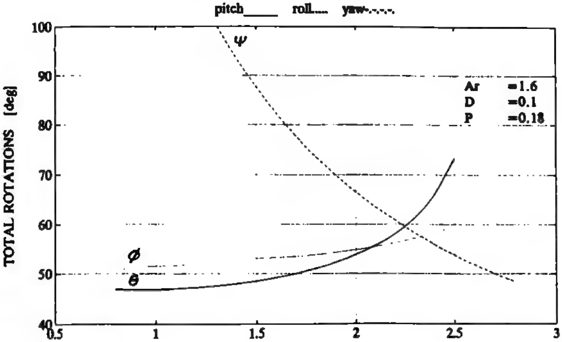
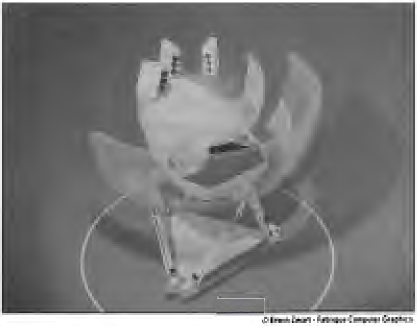


Figure 14(b). Pitch, Roll, Yaw. Maximum Total Translations as a function of lower gimbal circle radius



© [illegible] Computer Graphics

Figure 15. Artist's impression of SIMONA Research Simulator

AIAA-96-3478-CP

PHASE RESPONSE REQUIREMENTS BETWEEN CROSS-COUPLED MOTION AXES FOR HANDLING QUALITIES RESEARCH SIMULATORS

William W. Chung*, Ron Gerdes†, and Soren LaForce††
NASA Ames Research Center
Moffett Field, California

Abstract

An experiment was designed to study the effect of discrepancies between cross-coupled motion axes responses and to determine if phase response requirements for motion simulators is necessary for handling qualities research. Since the pilot is generally not located at the rotational center of a motion platform, this can produce distorted cues in a motion based flight simulator. For kinetically cross-coupled motion axes, such as roll and lateral, coordinated translational commands are required in order to compensate for induced linear accelerations caused by angular motion at the pilot station. The effect of phase response discrepancies between the roll and lateral cross-coupled motion axes was the focus of this experiment. A stability derivative model, which represents a fully decoupled aircraft response, was tailored to meet Level I rotorcraft handling qualities at low speed. The Vertical Motion Simulator, a gimbaled motion system at NASA Ames Research Center, provided the six degrees of freedom motion. The roll and lateral motion dynamics were modified to produce specific phase characteristics. This was required to study phase requirements between the roll angular acceleration and lateral specific force. Two low speed tasks, hover and side step, were used to evaluate the vehicle's handling qualities under three motion configurations and two visual delay configurations. The phase characteristics between the roll and lateral motion axes as well as the phase characteristics of the visual system were varied in the experiment. The test results indicated significant reduction in pilot workload and improved performance when the cross-coupled motion axes were in phase with each other and with the visual responses. A mismatch of phase response in the cross-coupled motion axes, up to 40 msec phase difference, led to increased pilot workload and poorer pilot handling qualities ratings in most instances. Due to a resulting large phase discrepancy between the visual and motion cues, the results also suggest that visual delay compensation had little or no effect on pilots' handling qualities ratings under the given test conditions.

NOMENCLATURE

δ_c	pilot collective stick input, in.
δ_{lat}	pilot lateral stick input, in.
δ_{lon}	pilot longitudinal stick input, in.
δ_r	rudder pedal input, in.
ϕ	roll attitude, rad
$\varphi_{\omega f}^2$	mean-square-value over the specified frequency spectrum, n.d.
θ	pitch attitude, rad
τ	fitted time delay for visual and motion response, sec
ω_m	simulator angular rate vector, rad/sec,
$\dot{\omega}_m$	simulator angular acceleration vector, rad/sec^2
$\dot{\omega}_{pilot}$	helicopter angular acceleration vector, rad/sec^2
$\dot{\omega}_{m\_cmd}$	simulator angular acceleration command vector, rad/sec^2
$\mathbf{a}_{mp}$	linear accelerations generated by the motion simulator at the pilot station, ft/sec^2
$\mathbf{a}_{ps}$	total linear accelerations sensed by the pilot, ft/sec^2
$\mathbf{a}_{pilot}$	helicopter pilot station acceleration vector, ft/sec^2
$\mathbf{a}_{rc}$	simulator rotational center (RC) acceleration vector, ft/sec^2
$\mathbf{a}_{rc\_cmd}$	simulator rotational center (RC) acceleration command vector, ft/sec^2
$\mathbf{g}$	gravitational vector, ft/sec^2
H(s)	fitted linear transfer function of visual and motion response without the time delay
$L_{\delta lat}$	roll control power, rad/sec^2/in.
L_p	roll damping coefficient, 1/sec
$M_{\delta lon}$	pitch control power, rad/sec^2/in.
M_q	pitch damping coefficient, 1/sec
$N_{\delta r}$	yaw control power, rad/sec^2/in.
N_r	yaw damping coefficient, 1/sec
p	helicopter model roll rate, body axis, rad/sec
$\dot{p}$	helicopter roll angular acceleration, rad/sec^2

* Aerospace Engineer. Member of AIAA.
† Engineering Test Pilot. LOGICON SYSCON Services, Inc.
†† Simulation Engineer. LOGICON SYSCON Services, Inc.

P(s)	a linear representation of visual and motion response with time delay
$\dot{p}_{cmd}$	roll acceleration motion command, rad/sec^2
$\dot{p}_m$	roll motion acceleration response, rad/sec^2
q	helicopter model pitch rate, body axis, rad/sec
$\dot{q}$	helicopter pitch angular acceleration, rad/sec^2
$\dot{q}_{cmd}$	pitch acceleration motion command, rad/sec^2
$\dot{q}_m$	pitch motion acceleration response, rad/sec^2
r	helicopter model yaw rate, body axis, rad/sec
$\dot{r}$	helicopter yaw angular acceleration, rad/sec^2
$\mathbf{r}$	position vector of the pilot station with respect to simulator RC, ft
$\dot{\mathbf{r}}$	relative velocity vector of the pilot station with respect to simulator RC, ft/sec
$\ddot{\mathbf{r}}$	relative pilot station linear acceleration with respect to simulator RC, ft/sec^2
$T_a(s)$	transfer function of angular motion axis
$\mathbf{T}_m$	direction cosine matrix from inertial to body axes of the simulator, n.d.
$\mathbf{T}_{mc}$	direction cosine matrix from inertial to simulator body axes attitude excluding the component used for low frequency linear specific force, n.d.
$T_t(s)$	transfer function of translational motion axis
u	helicopter model translational velocity, x-body axis, ft/sec
v	helicopter model translational velocity, y-body axis, ft/sec
w	helicopter model translational velocity, z-body axis, ft/sec
$W_a(s)$	transfer function of angular washout filter
$W_t(s)$	transfer function of translational washout filter
x_ε	longitudinal position error for hover and sidestep tasks, ft
X_u	longitudinal damping coefficient, 1/sec
y_ε	lateral position error for hover task, ft
y_{cg}	lateral c.g. position for sidestep task, ft
Y_v	lateral damping coefficient, 1/sec
$Z_{\delta c}$	vertical control power, ft/sec^2/in.
Z_w	vertical damping coefficient, 1/sec

Introduction

Motion simulators are widely used in handling qualities research and flight training. These applications depend on onset accelerations produced by the motion platform in combination with cues presented to the pilot from visual displays, control force feel, audio effects, and instrumentation displays. The fidelity of the onset accelerations is subject to the modeled aircraft dynamic characteristics, motion system's dynamic characteristics, motion control algorithms, and displacement constraints. For ground based motion simulators, this presents quite a challenge, because the displacement constraints dominate the motion fidelity issue. Washout filters are generally used in motion control logic to generate initial onset accelerations within the physical displacement constraints, i.e. the angular and translational limits. Therefore, the washout filters must be tuned to deliver consistent onset accelerations that complement the cues perceived by the pilot from other simulated devices.

To establish a direct correlation between simulation fidelity and handling qualities, Reference 1 suggests a criteria based on washout gains and phase characteristics as a measurement of motion cueing fidelity. Reference 2 follows the same frequency response approach and develops a 30 degree phase distortion criteria to compare perceived simulation cues for handling qualities evaluations. These were also the guidelines applied in developing the motion configurations for this experiment.

Reference 3 suggests that many motion cue errors are introduced in flight simulation due to physical constraints of motion platforms. Of all the motion cues perceived by the pilot, there is a fundamental element that is directly dependent on the kinetically cross-coupled motion system dynamic characteristics. This is a result of rigid body induced linear accelerations due to angular motion. Both Reference 4 and 5 indicate that translational accelerations sensed by the pilot are from the vestibular system and tactile mechanisms in the body. Due to the nature of human organ characteristics, lower frequency motion perception is sensed by the vestibular system and higher frequency motion perception is sensed by pressure from the pilot tactile mechanisms. Therefore, when the pilot station is not at the rotational center of the motion platform, an element of translational accelerations, i.e. induced linear accelerations, will be sensed by pilots due to angular motion. Induced linear accelerations are generally compensated in motion commands by assuming that the cross-coupled motion axis responses are the same. However, if the dynamic characteristics of two cross-coupled motion axes are not the same, or caused by different motion washout filter characteristics, discrepancies will be presented to the pilot and have an impact on the simulation fidelity. The objective of this experiment was to study the effect of phase differences between two kinetically cross-coupled motion axes and to determine if a requirement that defines acceptable phase discrepancy between the cross-coupled motion axes is necessary for ground based flight simulators.

Description of the Experiment

For a motion simulator where the pilot center of gravity is not at the rotational center of the motion platform, the specific force vector that is sensed by the pilot is governed by equation 1.

$$\mathbf{a}_{ps} = \mathbf{a}_{mp} - \mathbf{T}_m \cdot \mathbf{g} \qquad (1)$$

The accelerations produced by the motion system at the pilot station, $\mathbf{a}_{mp}$, is defined by equation 2.

$$\mathbf{a}_{mp} = \mathbf{a}_{rc} + \ddot{\mathbf{r}} + \dot{\omega}_m \times \mathbf{r} + \omega_m \times (\omega_m \times \mathbf{r}) + 2\, \omega_m \times \dot{\mathbf{r}} \qquad (2)$$

The position vector of the pilot, $\mathbf{r}$, is fixed relative to the rotational center, i.e. $\dot{\mathbf{r}}$ and $\ddot{\mathbf{r}}$ are zero. By assuming the rotational rate of the simulator cockpit, ω_m, is relatively small, equation 2 can be simplified to

$$\mathbf{a}_{mp} \approx \mathbf{a}_{rc} + \dot{\omega}_m \times \mathbf{r} \qquad (3)$$

The second term at the right hand side of equation 3 is the induced linear acceleration due to rotational motion, and the effect of this term is generally compensated in the motion commands such that this motion-platform-dependent term is not presented to the pilot. Similar reason also applies to the second term in equation 1 in compensating for the gravity component as a function of cab attitude. Therefore, the simulator translational and angular motion commands $\mathbf{a}_{rc\_cmd}$ and $\dot{\omega}_{s\_cmd}$ are defined as:

$$\mathbf{a}_{rc\_cmd} = W_t(s) \cdot \mathbf{a}_{pilot} - W_a(s) \cdot \dot{\omega}_{pilot} \times \mathbf{r} + W_a(s) \cdot \mathbf{T}_{mc} \cdot \mathbf{g} \qquad (4)$$

$$\dot{\omega}_{m\_cmd} = W_a(s) \cdot \dot{\omega}_{pilot} \qquad (5)$$

But the actual simulator responses from translational and angular motion commands are determined by the individual motion axis dynamic characteristics, given by,

$$\mathbf{a}_{rc} = T_t(s) \cdot \mathbf{a}_{rc\_cmd} \qquad (6)$$
$$\dot{\omega}_m = T_a(s) \cdot \dot{\omega}_{m\_cmd} \qquad (7)$$

Therefore, the perceived motion cues in kinetically cross-coupled axes are dependent on the dynamic characteristics of both the washout filters and the motion hardware. If the overall dynamic characteristics, such as the phase characteristics of the lateral and roll motion responses, are not the same, then pilots will be subjected to erroneous linear acceleration cueing.

Math Model Description

A mathematical model in stability derivative form was developed to represent the dynamic characteristics of a rate command helicopter[6] that is fully decoupled. The equations of motion are defined by equation 8 and 9.

$$\begin{bmatrix} \dot{u} \\ \dot{w} \\ \dot{q} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} X_u & 0 & 0 & -g \\ 0 & Z_w & 0 & 0 \\ 0 & 0 & M_q & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} u \\ w \\ q \\ \theta \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ Z_{\delta c} & 0 \\ 0 & M_{\delta e} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \delta_c \\ \delta_{lon} \end{bmatrix} \qquad (8)$$

$$\begin{bmatrix} \dot{p} \\ \dot{r} \\ \dot{v} \\ \phi \end{bmatrix} = \begin{bmatrix} L_p & 0 & 0 & 0 \\ 0 & N_r & 0 & 0 \\ 0 & 0 & Y_v & g \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} p \\ r \\ v \\ \phi \end{bmatrix} + \begin{bmatrix} L_{\delta_a} & 0 \\ 0 & N_{\delta_r} \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \delta_{lat} \\ \delta_r \end{bmatrix} \qquad (9)$$

Roll and pitch damping characteristics and control sensitivities are shown in Table 1. The aircraft characteristics were chosen to meet ADS-33D[7] Rotorcraft Level I handling qualities performance in low speed.

Visual System

An interchangeable cab used on the Vertical Motion Simulator (VMS) is equipped with a visual system that uses an ESIG-3000 image generator by Evans & Sutherland. The cockpit field of view is shown in Fig. 1. The visual system has a pure transport delay of 60 msec from pilot's input to a full screen image update. With this visual time delay, the bandwidth on the roll response is reduced from 10 rad/sec to 4.5 rad/sec, which still meets the Level I handling qualities requirements. A compensation algorithm[8] can be used to eliminate this visual delay, as was verified by Ref. 2. The visual time delay was one of the experimental parameters that was used to investigate the relationship between motion cross-coupled axes dynamic characteristics and visual delay characteristics.

Motion System

The VMS, as shown in Fig. 2, is a six degree-of-freedom motion platform that permits large excursions in the vertical and lateral axes. The vertical motion axis is driven by eight mechanically coupled 150-horsepower direct-current servomotors as outlined in Ref. 9. The lateral axis is driven by four 40-horsepower direct-current servomotors. Roll, pitch, yaw, and longitudinal are driven by four independent hydraulic systems with 2400 psi hydraulic pressure. The motion system's roll and lateral dynamic characteristics were tuned to three configurations for this experiment to study the effect of the phase difference between the two cross-coupled motion axes. The lateral accelerations due to yaw motion were not present due to the fact that the pilot longitudinal c.g. position was near to the gimbals rotational center for this experiment.

Three motion configurations were developed by using the visual system's 60 msec time delay as a reference. The VMS visual and motion system responses were fitted in a form defined by equation 10 which consists a linear transfer function, H(s), and a time delay, τ.

$$P(s) = H(s)\, e^{-\tau s} \qquad (10)$$

The characteristics of the visual system and the roll and lateral motion systems, P(s), of the three motion configurations and their equivalent time delays are shown in Table 2. The equivalent time delay is defined as a pure time delay that matches the phase response of P(s) between .1 to 10 rad/sec. The frequency responses of these three motion configurations, i.e. acceleration output versus acceleration input, are shown in Fig. 3, 4, and 5. The first motion configuration, MC1, the matched visual and roll and lateral motion cueing, was developed such that both roll and lateral motion dynamic phase responses matched the visual phase response. The second motion configuration, MC2, delayed lateral motion, was developed to keep the roll axis phase response in phase with the visual system, but to delay the lateral axis phase response by 40 msec. The third motion configuration, MC3, delayed roll and lateral motion, was designed to keep the phase response of both roll and lateral motion axes 40 msec behind the visual response. The first configuration represents the best phase match of both visual response and roll-lateral motion response as perceived by the pilot. Dynamic response for each configuration was tuned to have a satisfactory phase response up to 10 rad/sec.

Motion Washout Filters

The VMS motion drive logic is shown in Fig. 6. Washout filters are applied to translational and rotational pilot station accelerations after being transferred to the inertial frame to keep the simulator within the physical travel limits. Turn coordination and induced acceleration compensation keep the cross-coupled motion commands in accordance to pilot position states relative to the rotational center. A low pass filter is used to tilt the cab in supplementing linear motion cueing at low frequency.

For the experiment, two motion washout configurations as shown in Fig. 7, were developed for the hover task to investigate the phase difference effect on pilots' handling qualities. The high fidelity configuration was developed to keep the phase of roll and lateral washout filters the same, i.e. $\phi(W_a(s)) = \phi(W_t(s))$, and to keep both angular and translational motion cueing within the high fidelity region according to Ref. 1. The mixed fidelity configuration represented a case investigated in Ref. 10.

For the sidestep task, the motion washout filters were configured as shown in Fig. 8. Again, the washout filter frequencies for roll and lateral axes were chosen to have the same phase characteristics.

The dynamic characteristics of the rotorcraft and perceived visual and motion cueing under each motion washout and motion dynamic configuration (with the visual time delay of 60 msec) are shown in Fig. 9 to 11. The acceptable fidelity range for the high fidelity washout configuration, based on the 30 degree phase distortion criteria from Ref. 2, is summarized in Table 3 for all three motion configurations. The acceptable fidelity range is defined as the frequency spectrum where the phase difference between perceived visual cueing and motion cueing is less than 30 degrees. The acceptable fidelity range for the same group of motion configurations but with a visual compensation of 60 msec are shown in the same table to present the effect of the improved pilot perceived model response. As shown with the visual compensation, the pilot perceived an improved roll model response from a bandwidth of 4.5 rad/sec to 10 rad/sec as defined by the math model. However, phase improvement in the visual cueing alone would also increase the discrepancy between perceived visual and motion responses. As a result, a more restricted lateral-directional acceptable fidelity range over the frequency spectrum was developed.

Tasks

Two low speed tasks, Hover and Sidestep, were developed following the guidelines from ADS-33D under the no wind condition. Portions of the task procedures were modified to match the procedures developed in Ref. 2.

For the Hover task, the pilot was positioned at an angle with respect to the designated hover point, outlined in Fig. 12. The helicopter was initialized at 15 ft altitude. The pilot was asked to translate to a hover position over the desired hover point, with a ground speed of 6 kts, while maintaining the altitude. The desired hover point was defined by a hover target with a sight to indicate lateral position and height cues and a color-coded wall at a 45 degree angle to define longitudinal position cues. The transition to the hover point was to be achieved in one smooth maneuver, i.e. a smooth acceleration command followed by a smooth deceleration command. Creeping up to the final position was not allowed. The time for the pilot to stabilize at the desired hover point, from initiation of deceleration control input, was 15 seconds. Once in a stabilized hover, the pilot was asked to maintain hover position for 30 seconds. Rotorcraft deviations were measured from the desired hover point to determine pilots performance with respect to specified performance criteria, as given in Table 4.

For the Sidestep task, starting from a stabilized hover with the longitudinal axis of the rotorcraft oriented 90 degrees to the runway, as shown in Fig. 13, the pilot was asked to initiate a rapid and aggressive lateral translation, with a bank angle of at least 20 degrees, holding altitude constant with power. When the rotorcraft achieved a lateral velocity within 5 knots of the maximum allowable lateral airspeed, 30 knots, the pilot immediately initiated an aggressive deceleration to hover at constant altitude. The peak bank angle during deceleration was kept to at least 20 degrees, and occurred just before the rotorcraft came to a stop. Longitudinal and vertical position deviations were measured against the desired performance criteria, as shown in Table 4.

The visual data base was developed to provide visual cues for each task. Pylons and walls were color-coded such that the pilot could easily identify desirable and adequate performance envelopes. At the end of each task, the pilot was asked to give a handling qualities rating (HQR) based on the Cooper-Harper scale of Ref. 11.

A modified sidestep task was developed during the experiment to better reveal the significance of phase characteristics of the model-to-motion response. A closed loop task was added at the end of the sidestep maneuver by asking the pilot to hover before a designated pylon, with the same desirable performance criteria defined as before. Due to time limitations, only one pilot examined the modified sidestep task, and no pilot HQR was taken.

Results

The effects of kinetically cross-coupled motion dynamics were analyzed by studying HQRs and comments. Pilot control stick response and task performance data were also evaluated. The summary of the results are as follows:

Hover with high fidelity washout configuration

Pilot HQRs for three motion configurations are shown in Fig. 14. In comparing the first two motion configurations, i.e. matched cueing response (MC1) versus delayed lateral motion (MC2) two pilots, A and B, noted coordinated roll-lateral motion cueing which allowed them make accurate lateral inputs and pay more attention to longitudinal position control under the matched cueing configuration. Pilot B rated the matched cueing configuration better than the delayed lateral motion configuration. Pilot A felt that the matched cueing case (MC1) provided more solid motion cueing relative to the visual response, which reduced his physical and mental workload from that of the lagged lateral case. The increase in physical workload is strongly supported by the representative pilot lateral stick power spectral density (PSD) plot, given in Fig. 15, and the time trace of the pilot stick motion during the position-holding part of the task, Fig. 16. The power spectral density is the normalized energy distribution across the frequency spectrum. These data clearly showed that pilot workload associated with the lateral controller was reduced significantly across the frequency spectrum in the matched visual and motion cueing configuration. However, according to pilot A, the noted improvements in roll-lateral motion cueing response did not outweigh the required workload to hold the longitudinal position. Pilot C felt that both configurations required moderate pilot compensation to meet satisfactory performance criteria. He also felt that the delayed lateral motion had a slight advantage in pilot workload over the well matched case. For the delayed lateral motion configuration, jerkiness was among the common comments shared by all pilots.

The third motion configuration, motion lagged visual, was rated by two pilots, A and B. Pilot A rated this configuration worse than the matched cueing case and pilot B rated these two configurations with the same rating. Since the phase characteristics of the roll and lateral motion axes were the same in both configurations, the difference in pilot ratings could only result from the pilots' cueing preference, i.e. between the visual cueing and the motion cueing. Pilot A noted that some motion cues were lagging while pilot B noted that visual and motion cues were in harmony.

A summary of pilot lateral control mean-square-value (φ^2) and pilot cutoff frequency (ω_c) from PSDs developed by using CIFER, Ref. 12, is shown in Table 5. The pilot cutoff frequency approach is developed in Ref. 13 to compare pilot response characteristics under both flight and simulation conditions. By assuming a first order pilot response model, pilot cutoff frequency is defined as the frequency at half the power point of the total power spectral density of the given pilot control input, i.e. $\varphi_c^2 / \varphi_{total}^2 = 0.5$. The mean-square-value of the control with respect to the frequency spectra from 0 to ω_f, $\varphi_{\omega f}^2$, is equal to the total area under the PSD plot and is defined by equation 11,

$$\varphi_{\omega_f}{}^2 = \frac{1}{2\pi}\int_0^{\omega_f} G_{\delta\delta}\, d\omega \qquad (11)$$

where $G_{\delta\delta}$ contains the control power content as a function of frequency. Table 5 shows that under the matched cueing case, the total energy of the lateral control stick input consistently stays low among pilots in comparison with the other two mismatched conditions, which show comparable pilot cutoff frequencies.

Standard deviations of longitudinal and lateral position holding errors are given in Table 6. This table shows that pilots were able to maintain about the same level of performance regardless the test configurations, i.e. the change of motion parameters appeared to only affect the workload.

The longitudinal position cues were provided by the color-coded wall on the side window when in the stabilized hover position. Nonetheless, it did not provide an adequate range cueing sensitivity. This visual cueing deficiency combined with poorly coordinated pitch and surge dynamic characteristics with respect to visual cueing, Fig. 17, kept pilots' workload high in keeping longitudinal position within the satisfactory performance criteria, and made it more difficult in achieving Level I handling qualities performance.

Hover with mixed fidelity washout configuration

Pilot HQRs are shown in Fig. 14. The mixed fidelity motion configuration had a deviation in washout frequency between roll and lateral, 0.1 and 0.6 rad/sec respectively versus 0.3 for both axis in the high fidelity motion washout configuration. The washout gain on the lateral axis was also reduced from 0.9 to 0.4 in the mixed fidelity washout configuration. Roll washout gain was kept the same as the high fidelity washout case. The perceived roll and lateral motion cueing discrepancies as shown in Fig. 18 to 20, are much more significant at the low frequency range than in the high fidelity washout configuration. For pilot B and C, who evaluated these tasks, both felt that the matched case had much better coordinated motion cueing than the other two cases. The pilot comments were very similar to those in the high fidelity motion configuration. The workload for the matched configuration again showed reduced lateral control energy by both pilots, as given in Fig. 21. A summary of pilot cutoff frequency is shown in Table 5. It is noted that from the PSD data, and pilot comments, that there is no significant difference between the high fidelity and mixed fidelity motion configurations. The large phase discrepancy between roll and lateral motion at low frequency did not have a significant effect on pilot workload, or on performance. The phase discrepancy effect in high frequency, however, had a definite effect on pilot workload.

Pilot B's HQR was consistent with the result from Ref. 10. Pilot A evaluated all three motion configurations in mixed fidelity configuration. However, his data was contaminated with an incorrect washout filter setup. Therefore, no conclusion can be drawn to confirm the consistency between the experiments.

Sidestep

Pilot HQRs for the sidestep task are shown in Fig. 22. There is no clear trend to indicate the effect of cross-coupled motion dynamic response. The results for this task were hampered by a lack of range cues when the pilot proceeded to a hover stop. The lack of longitudinal position information, lightly damped pitch motion characteristics, and visual-motion phase discrepancies again led to an appreciable amount of pilot effort in stabilizing the helicopter within desirable performance criteria.

For the modified closed-loop sidestep task, only one pilot data point was taken to evaluate two motion configurations, i.e. the matched cueing and delayed lateral motion cases, without taking any HQR. The time traces of the control stick and position error from deceleration to a stabilized hover are shown in Fig. 23. The power spectrum of the lateral stick is shown in Fig. 24. The power spectral density of lateral stick and the pilot cutoff frequency are shown in Table 5. The PSD did not show any significant differences between the two motion configurations. However, pilot A commented that overall control felt solid without any overshoot tendency in the matched cueing configuration. Desirable performance was easily achieved. With lagged lateral motion, however, it was harder to stabilize, and there was a tendency to overshoot. This is shown in the position error time trace, given in Fig. 23. The motion in the latter configuration "felt jerky and artificial". It also required at least moderate pilot compensation to achieve desired performance, which would be a Level 2 handling qualities rating.

Visual Delay

HQRs from pilot A and B with visual delay compensation turned on and off are shown in Fig. 25. From both pilots' HQR on two washout configurations and three motion dynamic configurations, there is no significant difference in their ratings with and without the visual delay.

This result suggests that the improved model bandwidth response by removing the visual delay from the system was offset by the phase discrepancy between visual and motion cueing. Cueing discrepancies over the acceptable frequency range (Table 3) requires the pilot to mentally cross check the overall sensed model response, which meant increased pilot workload. The 30 degree phase distortion criteria provides a credible rationale for such a result.

Conclusions

A piloted motion based handling qualities flight simulation experiment was conducted to evaluate the significance of kinetically cross-coupled motion dynamic characteristics. Roll and lateral motion dynamic characteristics were perturbed for both precision hover and sidestep tasks. Visual delay and visual compensation were also evaluated under the same test conditions.

From pilot workload data, the phase characteristics of cross-coupled roll and lateral motion cueing has a significant effect on overall handling qualities of given tasks. Therefore, a requirement on cross-coupled motion axes phase characteristics with respect to visual response is strongly recommended to ensure the fidelity of flight simulation. The data from this experiment suggest that the roll dynamic response from motion cueing should at least match the visual response. The phase lag in lateral motion response with respect to the roll motion response should not be larger than 40 msec. Further investigations are required to define the specific phase criteria associated with the cross-coupled motion dynamic characteristics.

Visual delay compensation theoretically improves the simulation visual cueing responses, which should lead to better control bandwidth responses as well . Under the given test conditions, no noticeable pilot HQR or task performance improvement was found. That leads to the conclusion that the model response improvement made by visual cueing alone must be lost in the discrepancy between visual cueing and motion cueing. However, without the visual delay compensation, the vehicle's response characteristics is effectively reduced due to the inherited time delay in the digital flight simulation.

References

[1] Sinacori, J. B..: The Determination of Some Requirements for a Helicopter Flight Research Simulation Facility. Contractor Report STI-TR-1097-1, September 1977.

[2] Mitchell, D. G.; and Hart, D. C.: Effects of Simulator Motion and Visual Characteristics on Rotorcraft Handling Qualities Evaluations. American Helicopter Society Conference on Piloting Vertical Flight Aircraft, Jan. 1993.

[3] Grant, P. R.; and Reid, Lloyd D: Motion Washout Filter Tuning: Rules and Requirements. AIAA Flight Simulation Technologies Conference, August 1995.

[4] Schroeder, J. A.; and Johnson, W. W.: Yaw Motion Cues in Helicopter Simulation. Proceedings of the AGARD Flight Vehicle Integration Panel Symposium on Flight Simulation, May 1995.

[5] Gum, D. R.: Modeling of the Human Force and Motion-Sensing Mechanisms. Air Force Human Resources Laboratory, AFHRL-TR-72-54, June 1973.

[6] Lewis, M. S..; Mansur, M. H..; Chen, R. T. N.: A Piloted Simulation of Helicopter Air Combat to Investigate Effects of Variations in Selected Performance and Control Response Characteristics. NASA TM-89438, 1987.

[7] Aeronautical Design Standard, Handling Qualities Requirements for Military Rotorcraft. ADS-33D, July 1994.

[8] McFarland, R. E.: Transport Delay Compensation for Computer-Generated Imagery Systems.. NASA TM 100084, Jan. 1988.

[9] Danek, G. L.: Vertical Motion Simulator Familiarization Guide. NASA TM-103923, May 1993.

[10] Hart, D. C.; and Mitchell, D. G.:A Simulation Investigation of Motion Cueing and Visual Time Delay Effects on Two Helicopter Tasks. NASA TM 110385, April 1996.

[11] Cooper, G. E.; and Harper, R. P., Jr.: The Use of Pilot Rating in the Evaluation of Aircraft Handling Qualities. NASA TN D-5153, Apr. 1969.

[12] Tischler, M. B.; and Cauffman, M. G.: Frequency Response Method for Rotorcraft System Identification: Flight Applications to the BO-105 Coupled Rotor/Fuselage Dynamics. Journal of American Helicopter Society, Vol. 37, No. 3, pp. 3-17, July 1992.

Table 1. Damping characteristics and control sensitivity

X_u (1/sec)	Z_w (1/sec)	M_q (1/sec)	L_p (1/sec)	Y_v (1/sec)	N_r (1/sec)
0.0	-0.7	-4.3	-10.5	-0.12	-2.0
Z_{δ_c} (ft/sec^2/in)	$M_{\delta e}$ (rad/sec^2/in)	$L_{\delta a}$ (rad/sec^2/in)	N_{δ_r} (rad/sec^2/in)		
-9.873	0.45	1.8	0.04		

Table 2. Fitted VMS visual and roll-lateral motion response model, and equivalent time delay

		Fitted model response, P(s)	Equivalent time delay, msec
Motion configuration	Visual	$e^{-0.060s}$	60
1. Well matched visual and motion	Roll	$\frac{77.9}{s+80} e^{-0.052s}$	65
	Lateral	$\frac{2.39(152.4)(s^2+12s+94)}{(s^2+21s+225)(s^2+16.2s+164.5)} e^{-0.01s}$	68
2. Delayed lateral motion	Roll	$\frac{77.9}{s+80} e^{-0.052s}$	65
	Lateral	$\frac{152.4}{s^2 + 16.2 s + 164.5} e^{-0.01s}$	108
3. Delayed roll-lateral motion	Roll	$\frac{19.75}{s+20} e^{-0.072s}$	107
	Lateral	$\frac{152.4}{s^2+16.2s+164.5} e^{-0.01s}$	108

Table 3. Acceptable simulation fidelity range for high fidelity washout filter configuration, rad/sec

	With visual delay				With visual compensation			
Motion	Roll axis		Lateral axis		Roll axis		Lateral axis	
configuration	Min	Max	Min	Max	Min	Max	Min	Max
1. matched visual and roll-lateral motion	0.8	>60	0.8	17	0.75	9	0.75	9
2. delayed lateral motion	0.8	>60	0.8	7.8	0.75	9	0.75	5
3. delayed roll and lateral motion	0.8	8.5	0.8	7.8	0.75	5	0.75	5

Table 4. Task performance criteria

Task	Position Tolerance (ft)		Altitude Tolerance (ft)		Heading Tolerance (deg)		Time to Complete (sec)	
	D	A	D	A	D	A	D	A
Hover	± 3	± 8	± 2	± 4	± 5	±10	≤ 15	≤ 30
Sidestep	± 20	± 50	± 10	± 15	± 10	± 15		

Table 5. Pilot lateral stick power spectrum density and pilot cut-off frequency for hover task

Pilot	Motion configuration	High fidelity washout		Mixed fidelity washout		Modified sidestep	
		$\varphi_{\omega f}^2$	ω_c (rad/sec)	$\varphi_{\omega f}^2$	ω_c (rad/sec)	$\varphi_{\omega f}^2$	ω_c (rad/sec)
A	1. matched visual and roll-lateral motion	0.004	2.1			0.48	1.7
	2. delayed lateral motion	0.013	1.7			0.88	1.4
	3. delayed roll and lateral motion	0.015	2.4				
B	1. matched visual and roll-lateral motion	0.017	2.1	0.005	2.2		
	2. delayed lateral motion	0.055	2.4	0.059	2.1		
	3. delayed roll and lateral motion	0.065	1.8	0.037	1.9		
C	1. matched visual and roll-lateral motion	0.04	2.5	0.06	2.7		
	2. delayed lateral motion	0.051	2.4	0.037	3.3		
	3. delayed roll and lateral motion			0.055	4.1		

Table 6. Hover performance data with high fidelity washout configuration

		Longitudinal position error (ft)				Lateral position error(ft)			
Pilot	Motion configuration	Average	$\sigma_{x\varepsilon}$	Max	Min	Average	$\sigma_{y\varepsilon}$	Max	Min
A	1. matched visual and roll-lateral motion	-0.21	0.64	1.26	-1.08	0.1	0.44	1.27	-0.68
	2. delayed lateral motion	0.8	0.89	2.53	-1.12	0.24	0.41	1.26	-0.79
	3. delayed roll and lateral motion	0.9	0.88	2.53	-1.4	0.32	0.47	1.26	-0.80
B	1. matched visual and roll-lateral motion	0.03	1.34	2.04	-3.13	-0.04	0.71	1.0	-2.2
	2. delayed lateral motion	0.04	1.39	3.1	-2.42	-0.3	0.67	1.35	-1.32
	3. delayed roll and lateral motion	0.93	1.55	3.1	-3.1	0.81	0.67	2.4	-0.54
C	1. matched visual and roll-lateral motion	0.11	1.02	2.67	-1.72	-0.35	0.46	0.66	-1.57
	2. delayed lateral motion	0.17	0.98	1.26	-2.47	-0.03	0.57	1.15	-1.51
	3. delayed roll and lateral motion								

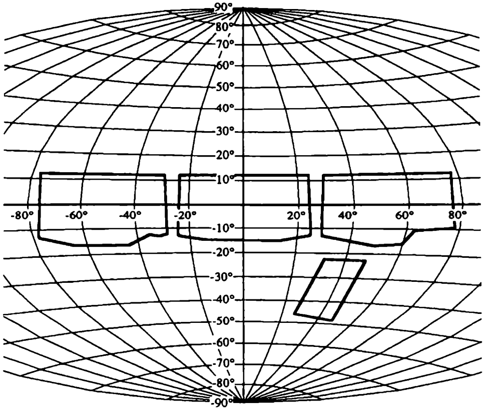
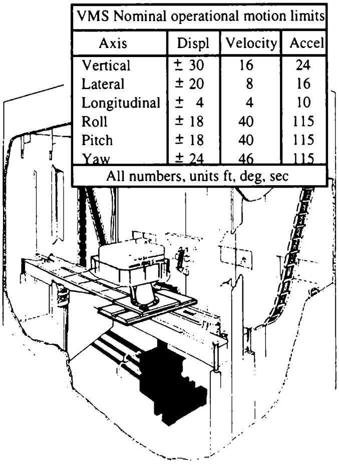


Figure 1. VMS R-cab display field of view



VMS Nominal operational motion limits			
Axis	Displ	Velocity	Accel
Vertical	± 30	16	24
Lateral	± 20	8	16
Longitudinal	± 4	4	10
Roll	± 18	40	115
Pitch	± 18	40	115
Yaw	± 24	46	115
All numbers, units ft, deg, sec			

Figure 2. NASA Ames Research Center VMS Vertical Motion Simulator

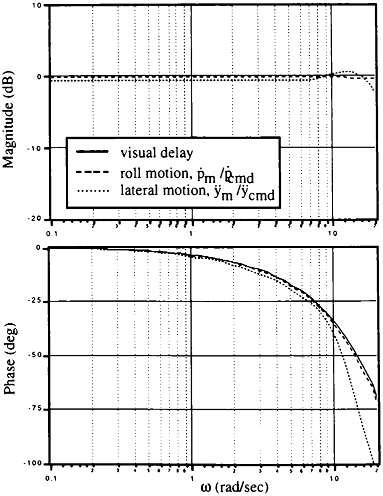


Figure 3. Matched visual, and roll and lateral motion configuration, MC1

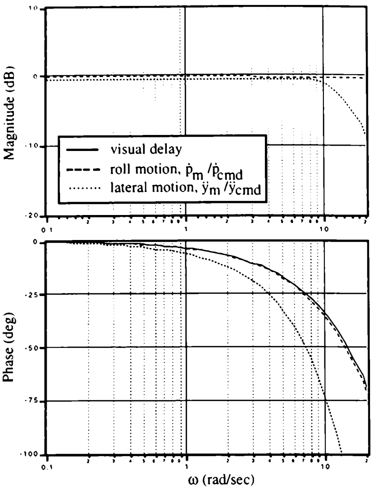


Figure 4. Delayed lateral motion configuration, MC2

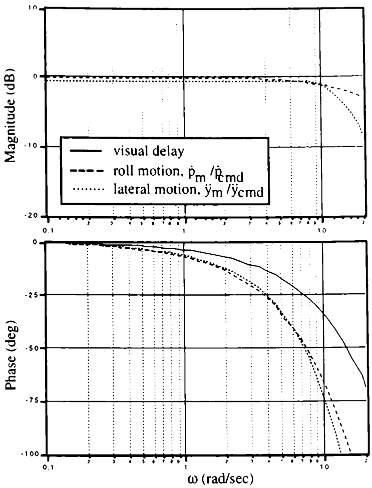


Figure 5. Delayed roll and lateral motion configuration, MC3

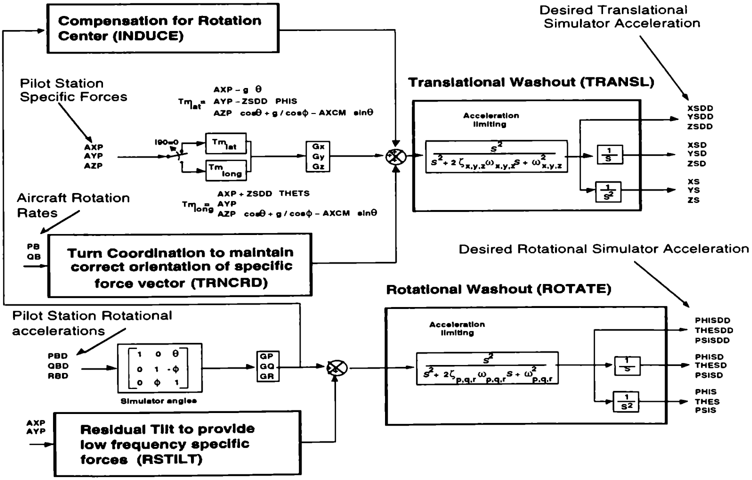

Compensation for Rotation Center (INDUCE)
Desired Translational Simulator Acceleration
Translational Washout (TRANSL)
Pilot Station Specific Forces
AXP
AYP
AZP
$Tm_{lat} = $ AXP - g θ / AYP - ZSDD PHIS / AZP cosθ + g / cosφ - AXCM sinθ
190=0
Tm_{lat}
Tm_{long}
Gx
Gy
Gz
Acceleration limiting
$\frac{s^2}{s^2 + 2\zeta_{x,y,z}\omega_{x,y,z}s + \omega^2_{x,y,z}}$
$\frac{1}{s}$
$\frac{1}{s^2}$
XSDD
YSDD
ZSDD
XSD
YSD
ZSD
XS
YS
ZS
Aircraft Rotation Rates
$Tm_{long} = $ AXP + ZSDD THETS / AYP / AZP cosθ + g / cosφ - AXCM sinθ
PB
QB
Turn Coordination to maintain correct orientation of specific force vector (TRNCRD)
Desired Rotational Simulator Acceleration
Rotational Washout (ROTATE)
Pilot Station Rotational accelerations
PBD
QBD
RBD
$\begin{bmatrix} 1 & 0 & \theta \\ 0 & 1 & -\phi \\ 0 & \phi & 1 \end{bmatrix}$
Simulator angles
GP
GQ
GR
Acceleration limiting
$\frac{s^2}{s^2 + 2\zeta_{p,q,r}\omega_{p,q,r}s + \omega^2_{p,q,r}}$
$\frac{1}{s}$
$\frac{1}{s^2}$
PHISDD
THESDD
PSISDD
PHISD
THESD
PSISD
PHIS
THES
PSIS
AXP
AYP
Residual Tilt to provide low frequency specific forces (RSTILT)

Figure 6. VMS motion drive commands block diagram

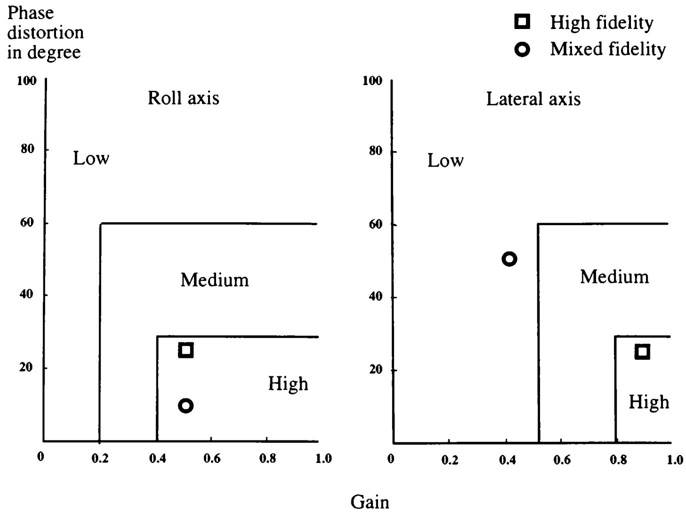


Figure 7. Motion cueing fidelity for hover task at 1 rad/sec

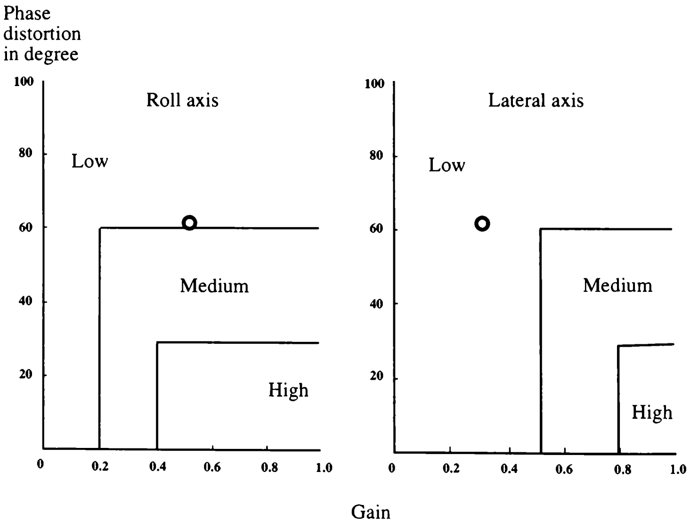


Figure 8. Motion cueing fidelity for sidestep task at 1 rad/sec

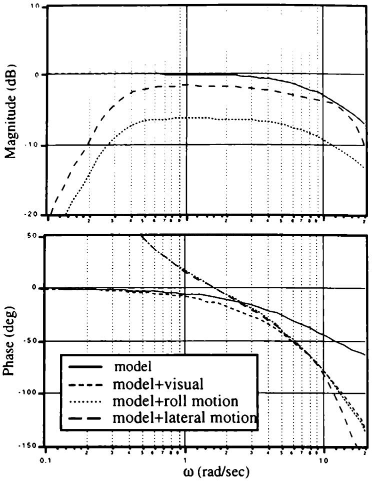


Figure 9. Roll model, and perceived visual and motion cueing responses in mathed cueing configuration, MC1

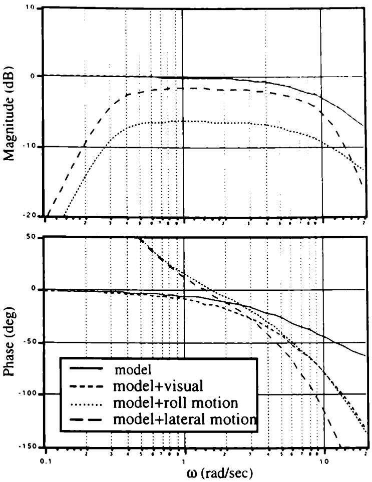


Figure 10. Roll model, and perceived visual and motion cueing responses in delayed lateral motion configuration, MC2

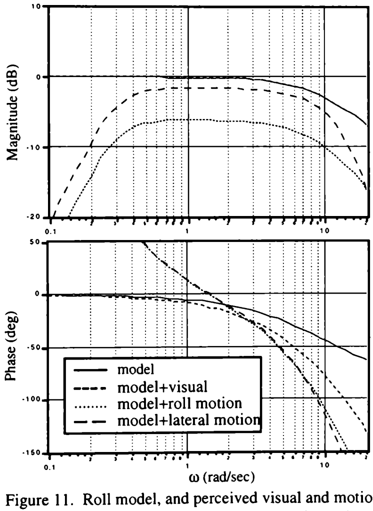


Figure 11. Roll model, and perceived visual and motion cueing responses in delayed motion configuration, MC3

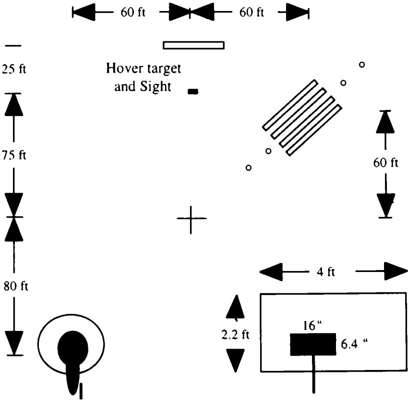


Figure 12. Hover task layout

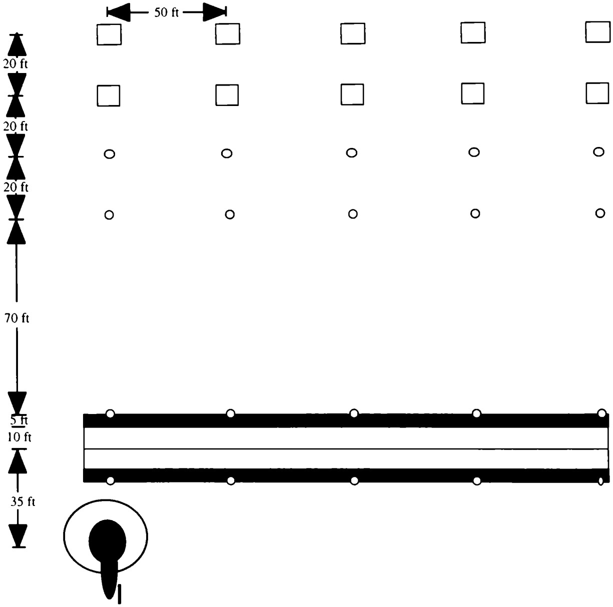


Figure 13. Sidestep task layout

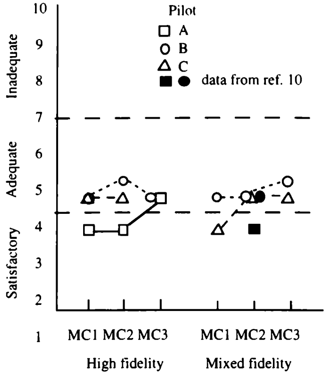


Figure 14. Pilot HQRs for hover task

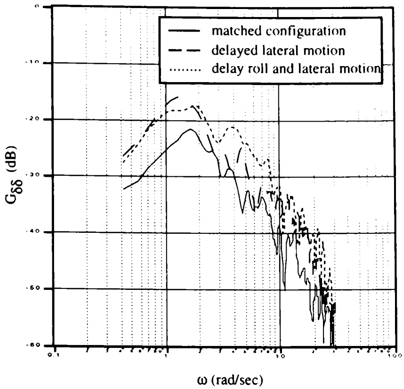
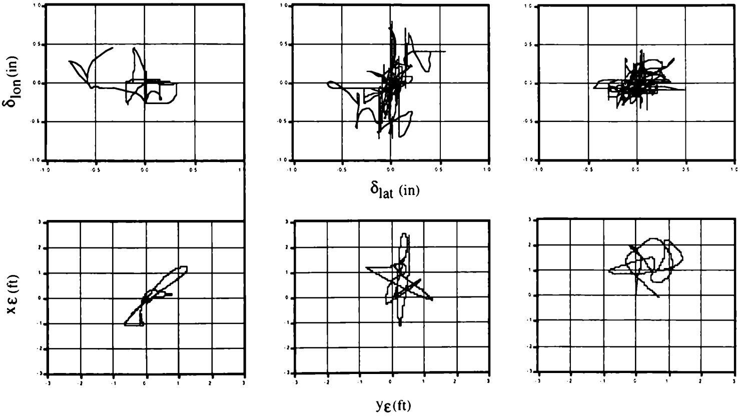


Figure 15. Pilot lateral contol stick power spectral density response for hover with high fidelity motion cueing configuration



Matched configuration
MC1

Delayed lateral motion
MC2

Delayed motion
MC3

Figure 16. Pilot control stick and position error time traces for the hover task

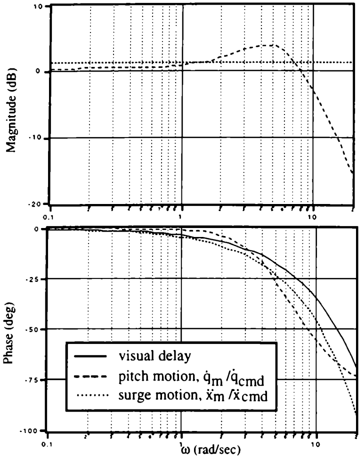


Figure 17. Visual delay, and pitch and surge response

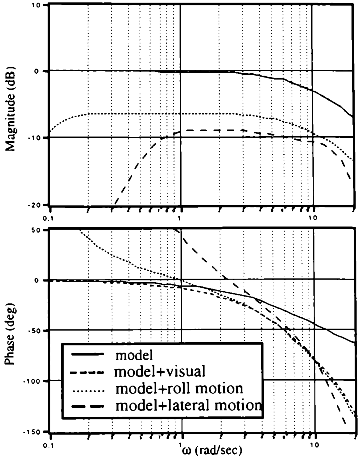


Figure 18. Roll model, and perceived visual and motion responses in matched, MC1, and mixed fidelity configuration

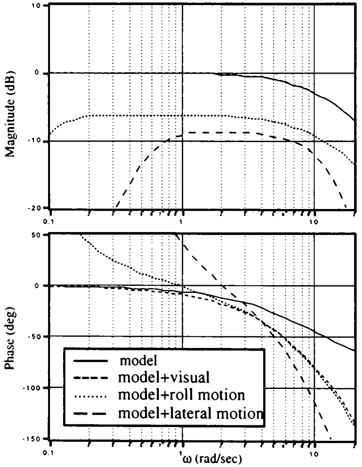


Figure 19. Roll model, and perceived visual and motion responses in delayed lateral motion, MC2, and mixed fidelity configuration

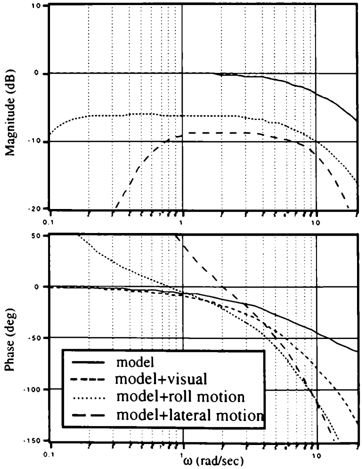


Figure 20. Roll model, and perceived visual and motion responses in delayed motion, MC3, and mixed fidelity configuration

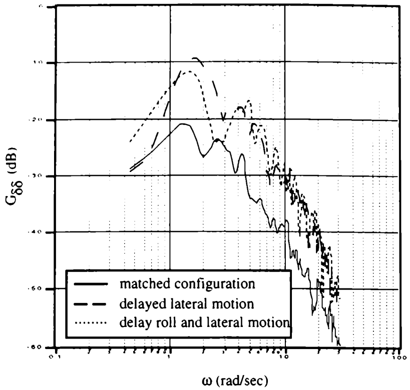


Figure 21. Pilot lateral contol stick power spectral density response for hover with mixed fidelity motion cueing configuration

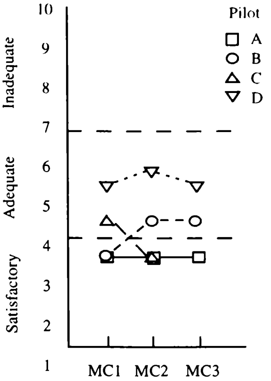
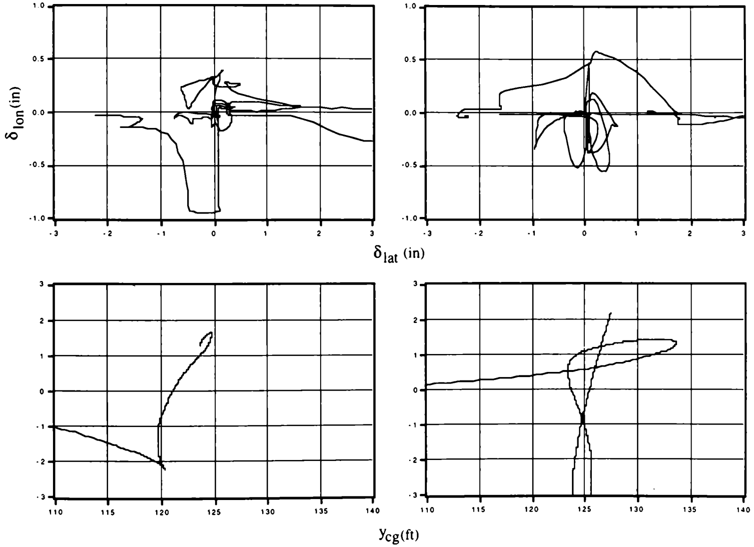


Figure 22. Pilots HQR for sidestep task with visual delay



Matched configuration, MC1

Delayed lateral motion, MC2

Figure 23. Pilot stick and hover position time traces for modified sidestep task

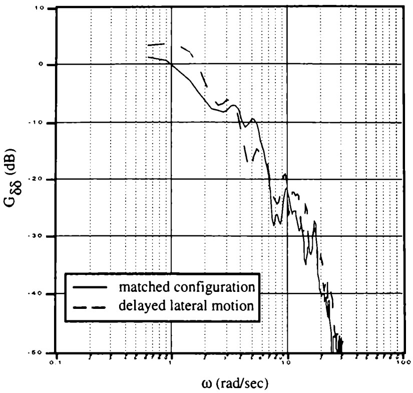


Figure 24. Pilot lateral contol stick power spectral density response for modified sidestep task

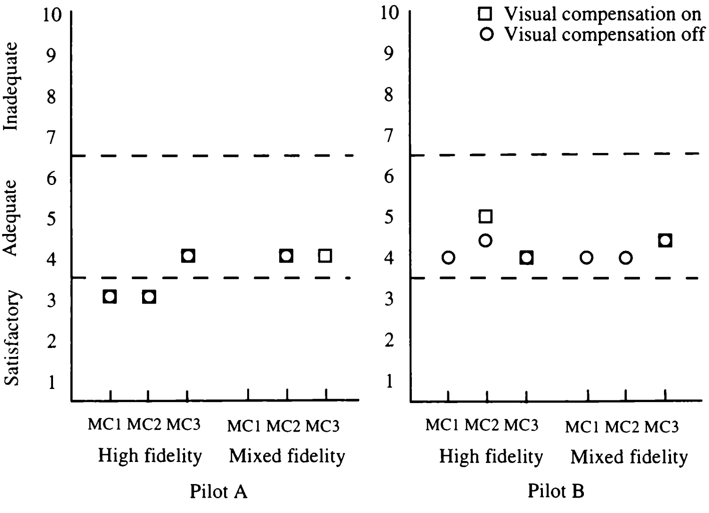


Figure 25. Pilot HQR with and without visual delay compensation

AIAA-96-3481-CP

A NEURAL NETWORK BASED APPROACH TO HELICOPTER LOW AIRSPEED AND SIDESLIP ANGLE ESTIMATION

Kelly M. McCool* David J. Haas*
Sea-Based Aviation Office
Carderock Division, Naval Surface Warfare Center
Bethesda, MD

Carl G. Schaefer, Jr.**
Aircraft Science and Technology Office
Naval Air Systems Command
Washington, D.C.

ABSTRACT

The objective of this study is to determine if a neural network based approach to helicopter low airspeed and wind direction estimation is feasible. Helicopter state parameters measured in the fuselage of a tandem rotor helicopter are used as inputs into two neural networks. One network is used to predict airspeed while the other classifies wind direction into one of four 90 deg quadrants. Results show that helicopter low airspeed estimation is sensitive to the influence of the ground proximity. For the out of ground effect flight condition, the backpropagation neural network estimates airspeed with an accuracy of approximately ±6 kt which is quite good since the uncertainty of the reference airspeed data is ±5 kt. Low airspeed estimation is more challenging in ground effect where network accuracy is ±10 kt. A linear vector quantization network is used to classify wind direction into forward, rearward, left sideward, and right sideward flight. The average successful classification rate out of ground effect is 95 percent (92, 100, 99, 90 for forward, rearward, and left and right sideward flight, respectively). In ground effect, the average successful classification rate is 89 percent (96, 94, 65, 100 for forward, rearward, and left and right sideward flight, respectively). These results suggest that a neural network based approach to low airspeed and wind direction estimation is quite promising.

INTRODUCTION

Military helicopters perform a wide variety of tasks including anti-submarine warfare, vertical replenishment, and search and rescue missions. These missions require that a large portion of flight time be conducted in the low speed flight regime below 50 kt. Unfortunately, in this regime instrumentation which measures airspeed and wind direction is generally lacking. This is due to inaccuracy of the traditional pitot static probe in the low speed environment as well as interference from the main rotor downwash. Although accurate low airspeed information is generally not available to the pilot, it is needed to maintain control margins since in the low speed regime increased engine power is required. Low airspeed information is also critical to accurate weapons firing solutions on attack helicopters.[1] In addition, high vibratory loads can occur in some low speed maneuvers which can result in fatigue damage accumulation in flight critical components. Technology to assist in monitoring the safe life remaining on these parts has been developed through helicopter usage monitoring and flight regime recognition techniques.[2-4] However, the success of this technology in the low speed regime is dependent on accurate airspeed information. Without correct low airspeed information these algorithms cannot recognize the low speed maneuvers and, therefore, may miss some fatigue accumulation data.

Development of a measurement system that accurately estimates low airspeed and wind direction has long been a difficult challenge. Interest in low airspeed measurement began in the 1950s when preliminary concepts were developed and flight tested.[5,6] These concepts involved mounting probes above the rotor hub as well as beneath the rotor in the wake. Since the 1950s, these concepts have been refined and a variety of low airspeed sensor designs have been flight tested. One such device is a swivel mounted apparatus installed above the rotor hub which can measure true airspeed magnitude and direction.[7,8] This system operates with two venturi tubes on opposite ends of a rotating arm. The differential pressure between the two sensors is used to calculate the airspeed and wind direction. This system, however, requires a slipring assembly or some alternative method of transferring the data from the rotating frame to the fuselage. Another approach involves a sensor designed to be mounted under the rotor and the nature of the

Presented at the AIAA Flight Simulation Technologies Conference, San Diego, CA, 29-31 July 1996,
AIAA 96-3481

* Aerospace Engineer

** Currently with FutureLink, Inc., Woodbridge, VA wake is used to determine helicopter airspeed.[9-13] This system uses a pitot-static probe which can rotate about 360 deg to provide airspeed and wind direction information. However, the flow environment under the rotor system is complex and empirical methods are used to linearize the output. Several other techniques including using ultrasonic transmission times and shed vortex characteristics have been proposed for deducing low airspeed and wind direction information.[1,14]

Unfortunately, many helicopters are not equipped with low airspeed sensors due to the complexity, expense, or the increased drag that these systems possess. Investment in this equipment is reserved for those aircraft with a critical low airspeed mission.

To date, only a few studies have been performed to analytically estimate airspeed in the low speed flight regime. Faulkner and Buchner[15] in 1980 used simplified thrust and flapping equations to obtain a forward and sideward velocity estimation. These equations are based on parameters which are measured in the fuselage but a knowledge of flap angle (a parameter which can only be measured in the rotating system) is also required. In 1986 Mandle[16] presented a similar technique for airspeed estimation based on the premise that true airspeed is related to collective and cyclic inputs. Promising results were achieved though the amount of correlation data were limited. A nonlinear state observer was developed by Mueller in 1987 to estimate helicopter airspeed.[17] However, the state observer approach is limited to forward flight.

The objective of the present study is to develop a neural network which uses helicopter state parameters measured in the fuselage to estimate low airspeed and sideslip angle. By using these parameters, the problem of moving a signal from the rotating system to the fuselage is eliminated. Furthermore, this approach could be easily implemented since the parameters chosen are quantities which are commonly measured on a flight data recorder. This method of low airspeed estimation provides for a mechanically simple, inexpensive alternative to current low airspeed measurement technology. The development of a neural network based approach to helicopter low airspeed estimation could also improve the performance of health and usage monitoring systems and simultaneously improve flight safety for those helicopters not equipped with low airspeed measurement systems.

TECHNICAL APPROACH

Database

The database used in this analysis is comprised of flight test data collected during a low airspeed test of a Navy CH-46 helicopter at the Naval Air Warfare Center, Aircraft Division, Patuxent River, MD. The vehicle was tested at airspeeds from hover to 50 kt over a full range of sideslip angles. Data were recorded during steady flight conditions as well as during accelerating forward flight starting from a hover. The reference airspeed was determined from the helicopter speed with reference to the ground which was measured with Doppler radar. Testing was performed only when prevailing winds were below 5 kt and since winds were not accounted for during the flight test the uncertainty in the reference airspeed data is at least ±5 kt.

The aircraft sideslip angle is derived from the reference Doppler velocities which are broken into two components, a forward and a sideward velocity component. The uncertainty of this reference sideslip angle is more significantly affected by prevailing winds particularly at low speeds. Sideslip angle uncertainty ranges from ±90 deg at 5 kt to ±5.4 deg at 50 kt. Table 1 shows the uncertainty in sideslip measurement as a function of reference airspeed for the data used in this study.

Reference Airspeed (kt)	Uncertainty in Reference Sideslip Angle Measurement (deg)
5	±90.0
10	±30.0
15	±19.5
20	±14.5
25	±11.5
30	±9.6
35	±8.2
40	±7.2
45	±6.4
50	±5.4

Table 1. Uncertainty of Reference Sideslip Angle Measurement.

Fifteen parameters measured in the fuselage during the flight test were chosen as inputs into the network and are listed in Table 2. These fifteen inputs are commonly monitored parameters and do not require mounting instrumentation in the rotating system.

1	Gross weight (lb)
2	Center of gravity (in)
3	Longitudinal cyclic stick position (%)
4	Lateral cyclic stick position (%)
5	Pedal position (%)
6	Collective stick position (%)
7	Pitch attitude (deg)
8	Roll attitude (deg)
9	Pitch rate (deg/sec)
10	Roll rate (deg/sec)
11	Yaw rate (deg/sec)
12	Engine torque #1 (ft lb)
13	Engine torque #2 (ft lb)
14	Rotor speed (rpm)
15	Altitude (ft)

Table 2. Neural network inputs.

Data Set Selection

Data set selection is a critical part of developing a successful neural network model. Two data sets are developed, one for training the neural network and one for testing the trained network. The training set should consist of data which fully represents the domain of the problem to be modeled. For this analysis the problem domain includes any airspeed and sideslip combination which might be encountered in the low speed environment. The training data set should not be weighted toward any one flight condition because the model will likely perform well in that condition but fail in other maneuvers. In addition, factors which might significantly affect the relationship between the inputs and the outputs must be considered. Faulkner and Buchner[15] identified that this relationship will change depending on whether the aircraft is in or out of ground effect. Difficulty in measuring low airspeed in ground effect has also been documented in flight test reports.[8,12]

These factors were taken into consideration while developing a training data set. First, the full database was separated into data corresponding to steady flight conditions and data taken during accelerating forward flight conditions. For the present study, only steady flight conditions are considered. In an attempt to develop a training data set which evenly represents the space of the low airspeed problem, a "binning" method was developed which involved separating the data into 36 sideslip ranges (10 deg intervals), 10 velocity ranges (5 kt intervals) and 3 gross weight ranges (high, medium, and low gross weight). This partitioned the data into a possible 1080 bins. However, the flight test did not cover every 1080 possible flight conditions and therefore not every bin contained data. The training set was developed by randomly selecting 12 data points from each bin. If a bin had less than 12 data points, then all of the points from that bin are selected.

The test data set consists of any data remaining in the 1080 bins after the training data are removed. The test data set is used to evaluate the network performance on data it was not trained on and thus provides a measure of how well the network generalizes.

In order to address the issue of whether being in ground effect or out of ground effect significantly affects network performance three separate training and test sets were developed. The first training and test set consists of all the steady flight conditions available and is called the baseline data. The second and third training and test sets are a subset of the baseline data and correspond to out of ground effect (OGE) and in ground effect (IGE) data. It is straight forward to identify when the helicopter may be influenced by ground effect by means of the pressure altitude measurement. The ground effect induces changes in static pressure which result in a negative pressure altitude readout. This serves as an indication of a ground effect condition.

Figures 1 and 2 show the flight conditions represented in the OGE and IGE training and test sets, respectively. The radial lines correspond to velocity in knots while the azimuthal lines correspond to sideslip angle. A zero degree sideslip angle corresponds to forward flight while 180 deg corresponds to rearward flight. A 90 or 270 deg sideslip angle corresponds to right or left sideward flight, respectively. Each point in Figures 1 and 2 corresponds to data taken at a rate of 10 Hz during a steady flight condition. There are 5582 OGE data points and 8770 IGE data points. From Figures 1 and 2, it can be seen that most rearward flight conditions were recorded in ground effect.

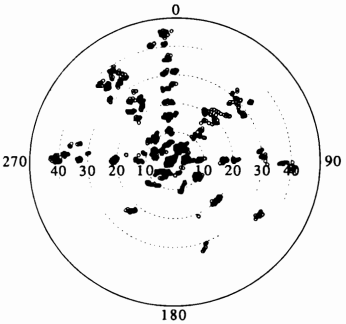


Fig. 1a. Out of Ground Effect Training Data Set.

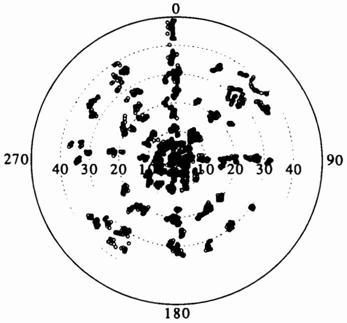


Fig. 2a. In Ground Effect Training Data Set.

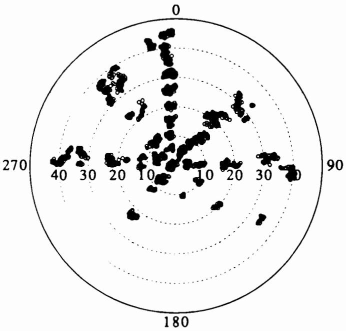


Fig. 1b. Out of Ground Effect Test Data Set.

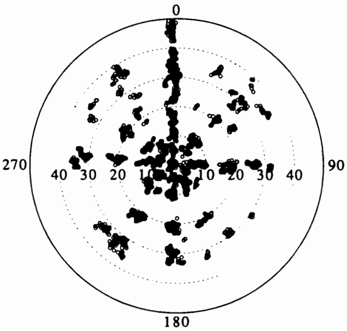


Fig. 2b. In Ground Effect Test Data Set.

Neural Network

Two types of networks are used in this study; a backpropagation network is used to predict helicopter airspeed and a linear vector quantization (LVQ) network is used to quantify sideslip angle. A typical backpropagation network consists of an input layer, one or more hidden layers, and an output layer. For example, Figure 3 shows a single hidden layer, 3 input, and 2 output backpropagation network architecture where each circle represents a processing element (PE). At each processing element a nonlinear transfer function with a connection weight is applied to develop a relationship between the input and output vectors.

For the backpropagation network in this analysis, the 15 inputs are listed in Table 2 and the single output is airspeed. The inputs are fed into the network and initially a random set of connection weights is applied. The resulting network output is compared with the desired output and the error is backpropagated through the connection weights which are adjusted appropriately. This process is iteratively repeated with new connection weights until the error between the calculated and desired outputs is minimized. An adaptive stepsize learning rule known as Extended Delta-Bar-Delta is used in the present analysis. This learning rule is more powerful than the standard

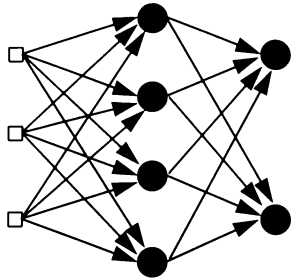


Fig 3. Schematic of typical backpropagation neural network architecture.

backpropagation learning rule since it trains more rapidly and the probability that the network will converge to a local minima is reduced.

To determine the optimal neural network architecture for low airspeed predictions, a parametric study was conducted on the OGE data set. The number of hidden layers and the number of PEs per layer was varied to determine which architecture best suits this problem. The statistical parameters used to assess the success of the model are the Pearson's correlation coefficient, R and root mean square (RMS) error of the test data set. Pearson's correlation coefficient is a measure of how linear a relationship is and RMS error provides a measure of the error. Performance of the neural network on the training data set will generally be quite good. Therefore, data that has not previously been presented (i.e. the test set) is used to test the network. The neural network was trained on the data for 640,000 iterations, that is, the entire training data set was passed through the network 375 times during the training process. Every 80,000 iterations the network was saved and evaluated using the test data set. When the RMS error for the test data set stabilized, the network was considered converged. This typically occurred around 500,000 iterations.

Results of this optimization study are shown in Table 3. A 2 hidden layer network with 25 PEs in each hidden layer was selected as the optimal network architecture since the RMS error for the out of ground effect test data set was small and the Pearson's correlation coefficient, R was very close to one. The 2 hidden layer 65 PEs per layer architecture produced a slightly smaller RMS error of 0.021 kt on the test data set. However, this improvement was not considered significant enough to warrant the added complexity of the additional processing elements.

Number of Hidden Layers	Number of PEs per Layer	Test Set RMS Error	Test Set R
1	25	3.440	.968
1	65	2.895	.976
2	15	2.842	.976
2	25	2.697	.979
2	65	2.676	.979

Table 3. Results of Network Architecture Study.

Sideslip angle is estimated with an LVQ classification network. The LVQ network consists of an input layer, a Kohonen layer, and an output layer as shown in Figure 4 for a 3 input, 2 output system. The LVQ network is a classification network which in this analysis is used to classify wind direction into one of four quadrants. The four network outputs are in the form of zeros and ones; {1,0,0,0} corresponds to a forward flight classification, {0,1,0,0} is a right sideward flight condition, and {0,0,1 ,0}, {0,0,0,1} are rearward and left sideward flight, respectively.

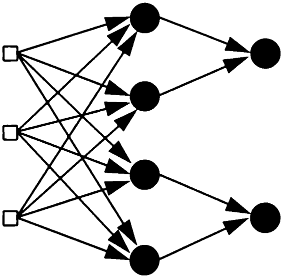


Fig. 4. Schematic of typical LVQ architecture.

The total number of PEs required in the Kohonen layer is data dependent. For this study, the number of Kohonen PEs corresponds to 10% of the number of training data points. There are 170 Kohonen PEs for the OGE network and 315 PEs for the IGE network. In the LVQ architecture, an equal number of Kohonen processing elements are assigned to an output. These groups of processing elements map only to their assigned output (see Figure 4). LVQ training occurs in two stages. In the first stage of training, the inputs are fed into the network which initially have a random set of connection weights. For each group of PEs a winner is established which most closely maps to the appropriate output. During training, the weights of the winning PEs are iteratively adjusted to improve the correlation. The second stage of training is a refinement which occurs after a reasonably good classification network is developed. In this stage misclassifications near the boundary between forward flight and left sideward flight, for example, are addressed until classification errors are minimized. During the two stages of training for this study, the training data set is passed through the network a total of 45 times before classification errors are minimized.

RESULTS AND DISCUSSION

Airspeed Predictions

Using the 15 parameters in Table 2 as neural network inputs and the optimal network architecture described in the previous section a method of predicting airspeed is developed. Results for the 2 hidden layer backpropagation network with 25 PEs per layer on the baseline test data set are shown in Figure 5. Figure 5b shows that the airspeed predictions are inadequate for the baseline test data set which includes in ground effect and out of ground effect data.

By dividing the data into two subsets, IGE and OGE data, it becomes evident that ground effect has a significant influence on the relationship between aircraft state parameters and airspeed. When the aircraft is out of ground effect the network results significantly improve as seen in Figure 6.

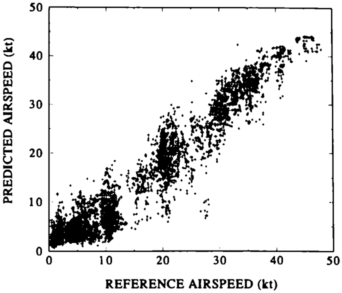


Fig. 5a. Airspeed predictions for the baseline training data set.

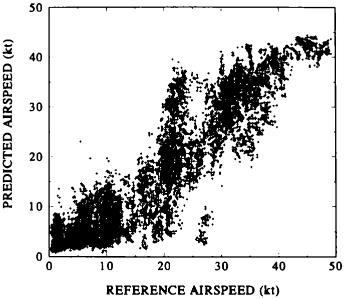


Fig. 5b. Airspeed predictions for the baseline test data set.

Figure 7 shows that modeling airspeed in ground effect is more challenging, particularly at airspeeds between 20 and 30 kt. This is not unexpected since the ground effect environment is quite complex with unsteady flow influencing pilot controls.

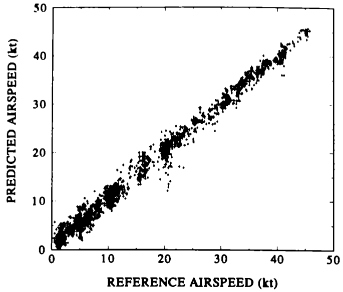


Fig. 6a. Airspeed predictions for the OGE training data set.

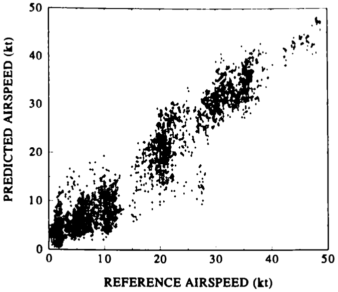


Fig. 7a. Airspeed predictions for the IGE training data set.

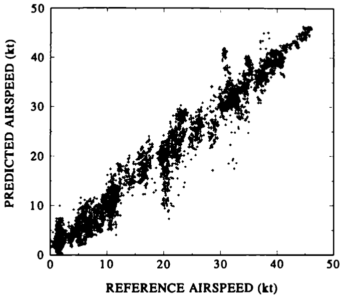


Fig. 6b. Airspeed predictions for the OGE test data set.

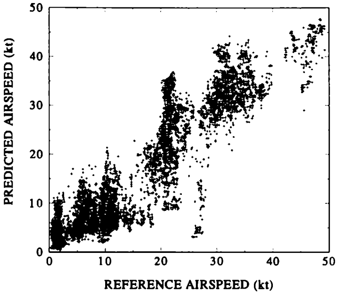


Fig. 7b. Airspeed predictions for the IGE test data set.

To examine the degree of nonlinearity between inputs and outputs, the network architecture with 2 hidden layers and 25 PEs per layer was modified by changing the nonlinear hyperbolic tangent transfer function of the PE to a linear transfer function. A linear transfer to the output was also applied. By using this technique, the network becomes equivalent to a linear regression analysis and the importance of nonlinearities in this problem can be assessed. Results for the linearized neural network are shown in Figure 8 for out of ground effect data. Comparing Figures 6 and 8, it is clear that the linear regression technique captures the general trends correctly but the nonlinear transfer functions of the neural network are required to significantly improve accuracy.

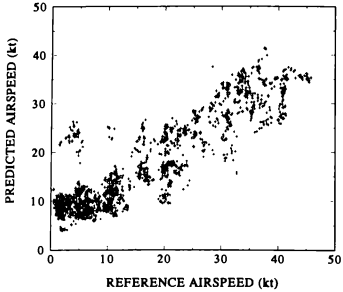


Fig. 8a. Airspeed predictions using a linear model (OGE training data set).

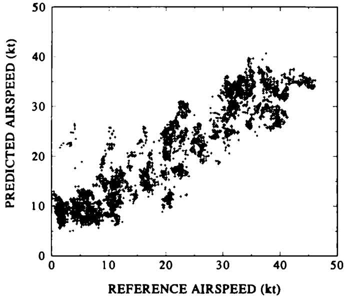


Fig. 8b. Airspeed predictions using a linear model (OGE test data set).

A statistical analysis of the nonlinear neural network error indicates that the error is close to a normal distribution with a mean of zero. Therefore, 95.5% of the data predictions will fall within ±2σ where σ is the RMS error. Table 4 shows the test data set results for the four cases examined.

	R	RMS error (kt)	Accuracy ±2σ (kt)
Nonlinear Model Baseline (IGE&OGE)	.92	±5.2	±10.4
OGE	.98	±2.7	±5.4
IGE	.92	±4.9	±9.8
Linear Model (Regression)	.92	±5.7	±11.4

Table 4. Summary of results (test data sets).

Sideslip Angle Classification

Sideslip angle was initially modeled using the same network architecture that was used for airspeed predictions, the two hidden layer network with 25 PEs in each layer. However, this architecture was not successful for sideslip angle predictions as can be seen in Figure 9 for the out of ground effect data set. One possible explanation for the lack of correlation with reference sideslip angle is that the uncertainty in this measurement is very high at airspeeds below 15 kt (see Table 1). To assess the effect of this uncertainty on sideslip angle prediction, all out of ground effect data with airspeeds below 15 kt were removed and the network was retrained. Unfortunately, this method did not improve accuracy. In fact, the correlation coefficient on the test data set decreased and RMS error increased.

Since the backpropagation network produced poor results when used to predict sideslip angle, an alternative approach was developed based on the classification of sideslip angle into four quadrants as shown in Figure 10. A forward flight classification corresponds to a sideslip angle of ±45 deg. Right and left sideward flight correspond to sideslip ranges of 45 to 135 deg and 225 to 315 deg, respectively. Rearward flight corresponds to a 135 to 225 deg sideslip angle. Table 5 presents results of this classification for OGE

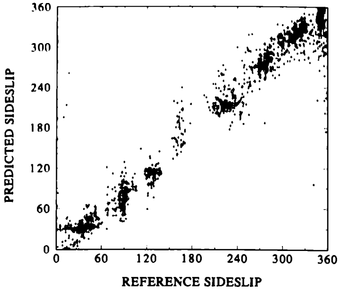


Fig. 9a. Sideslip angle prediction using 2 hidden layer backpropagation network with 25 PEs per layer (OGE training data set).

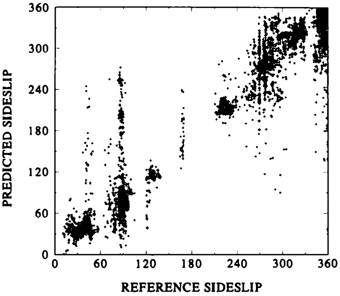


Fig. 9b. Sideslip angle prediction using 2 hidden layer backpropagation network with 25 PEs per layer (OGE test data set).

and IGE flight conditions. Successful classification rates are shown in Table 5 for the four sideslip angle quadrants. An average classification rate (AVG) is also calculated and is the average of the four quadrant rates. Results show that left sideward flight is a difficult maneuver to predict both in ground effect and out of ground effect.

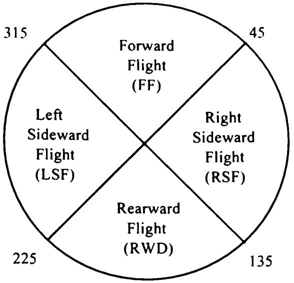


Fig. 10. Four quadrants of sideslip angle classification.

	FF	RSF	RWD	LSF	AVG
OGE					
Training	.98	.98	.99	.97	.98
Test	.94	.94	.89	.80	.89
IGE					
Training	.98	.99	.99	.98	.98
Test	.94	.95	.94	.72	.89

Table 5. Successful sideslip classification rates.

	FF	RSF	RWD	LSF	AVG
OGE					
Training	.99	.99	1.0	.99	.99
Test	.92	.90	1.0	.99	.95
IGE					
Training	.99	1.0	.99	1.0	1.0
Test	.96	1.0	.94	.65	.89

Table 6. Successful sideslip classification rate with reference airspeeds less than 15 kt removed.

Based on the premise that reference sideslip angle measurements made below 15 kt have a high degree of inaccuracy, all data with velocities less than 15 kt were removed and the LVQ network was retrained. This significantly improved classification of OGE data to an average successful classification of 95% on the test data set as shown in Table 6. However, IGE test results showed no net improvement. A possible reason for the difficulty in classifying the left sideward flight in ground effect condition is the nature of the tandem rotor configuration as shown in Figure 11. The forward rotor rotates counter-clockwise while the aft rotates clockwise. The rotor diameters overlap and on the left side tip vortices of the two rotors may converge and interact. In left sideward flight, these tip vortices may be blown further toward the fuselage and may influence the aircraft state.

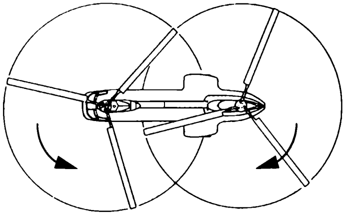


Fig. 11. Top view of CH-46 tandem rotor helicopter.

For the OGE condition, the difficulty in classifying left sideward flight was eliminated when velocities less than 15 kt were removed suggesting that larger left sideward flight velocities move the tip vortex interactions beyond the region of influence near the fuselage. However, when velocities less than 15 kt are removed for the IGE data set, left sideward flight classification got worse. This suggests that the lack of classification for the IGE left sideward flight condition is not due to the inaccuracy in the reference sideslip angle. It is most likely that the ground vortex caused by the wake impinging on the ground combined with the interacting tip vortices results in a dynamic unsteady flow region where it is difficult to estimate a steady sideslip angle with the network inputs used in this study.

SUMMARY AND CONCLUSIONS

The objective of this analysis was to determine if a neural network based approach to helicopter low airspeed and wind direction estimation is feasible. Results show that this approach is quite promising and would provide for a mechanically simple, inexpensive alternative to current low airspeed and sideslip angle measurement systems.

Accurate airspeed and sideslip angle estimation is highly dependent on whether the aircraft is flying in or out of ground effect. Estimates have the highest degree of accuracy for the out of ground effect flight condition. The unsteady flow environment present for the in ground effect condition provides an extra challenge for accurately predicting airspeed and sideslip angle.

Helicopter low airspeed is predicted with an accuracy of ±5.4 kt when the aircraft is out of ground effect. This accuracy is quite good given that the reference airspeed has an uncertainty of ±5 kt. Sideslip angle can be classified into one of four wind directions with a 95% classification accuracy when the aircraft is out of ground effect.

The left sideward flight in ground effect condition for a tandem rotor helicopter appears to be a difficult maneuver for sideslip angle classification. This may be due to a combination of the ground vortex and interacting rotor tip vortices being blown toward the fuselage causing an unsteady flow condition which influences the aircraft state.

A comparison of nonlinear network results with a linearized network (equivalent to a linear regression) show that the nonlinear nature of the neural network is required to achieve the necessary accuracy in airspeed prediction.

FUTURE WORK

An extension to this study is to analyze the accelerating forward flight maneuvers in the CH-46 low airspeed database. Methods also need to be identified to improve airspeed and sideslip angle estimates in ground effect. Low airspeed flight test data with more accurate measurements of airspeed and sideslip angle would be valuable for improving predictions and building confidence in this approach.

ACKNOWLEDGMENTS

The authors would like to acknowledge Mr. Glenn Saller of NAWCAD, Patuxent River, for providing flight test data and for technical discussions regarding the CH-46 low airspeed flight test.

REFERENCES

1. Ferrin, F.J. and Yurescko, M., "Ultrasonic Wind Vector Sensor," Presented at the Technical Conference on "The Effects of Helicopter Downwash on Free Projectiles," St. Louis, MO, August 1975.

2. Haas, D.J. and Schaefer, C.G., "Emerging Technologies for Rotor System Health Monitoring," Proceedings of the American Helicopter Society 52nd Annual Forum, Washington, D.C., June 1996.

3. Barndt, G.L. and Moon, S., "Development of a Fatigue Tracking Program for Navy Rotary Wing Aircraft," Proceedings of the American Helicopter Society 50th Annual Forum and Technology Display, Washington, D.C., May 1994.

4. Moon, S., Menon, D., and Barndt, G., "Fatigue Life Reliability Based on Measured Usage, Flight Loads, and Fatigue Strength Variation," Proceedings of the American Helicopter Society 52nd Annual Forum, Washington, D.C., June 1996.

5. Harbach, A.B., "Helicopter Low-Air-Speed Measurement," Bell Aerospace Co., Report no. 02-983-075, April 1954.

6. Skelly, E.T., "Modification and Flight Test of True Airspeed Indicator for Helicopters," CAL-IH-1232-F-1, Feb. 1959.

7. Green, D.L., "Review of LORAS Characteristics and Development History," Presented at the Technical Conference on "The Effects of Helicopter Downwash of Free Projectiles", St. Louis, MO, August 1974.

8. Abbott, W.Y., Boirun, B.H., Hill, G.E., Tavares, E.J., "Flight Evaluation of Pacer Systems Low Range Airspeed System LORAS 1000," USAAEFA-75-17-17, May 1977.

9. Carter, J., "The Measurement of Helicopter Air Data Using a Swiveling Pitot-Static Pressure Probe," Presented at the Technical Conference on "The Effects of Helicopter Downwash of Free Projectiles", St. Louis, MO, August 1974.

10. Arajs, P., "Helicopter Air Data System (HADS) and Advanced Digital/Optical Control System (ADOCS)," Final report DAAK51-84-C-0001, March 1987.

11. Ferrell, K.R., Boirun, B.H., Hill, G.E., "Low Airspeed Sensor Location Tests AH-1G Helicopter," USAAEFA-75-19-1, Feb. 1977.

12. Dominick, F.L., Ferrell, K.R., O'Connor, J.C., "Flight Evaluation Elliott Dual-Axis Low Airspeed System (LASSIE II)," USAAEFA-71-30-6, Sept. 1975.

13. Kaletka, J., "Evaluation of the Helicopter Low Airspeed System LASSIE," Proceedings of the Seventh European Rotorcraft and Powered Lift Aircraft Forum, Garmisch-Partenkirchen, Germany, Sept. 1981.

14. Joy, D., "Airspeed and Direction Measurement by Vortex Detection," Proceedings of the Air Data Symposium, Monterey, CA, June 1976.

15. Faulkner, A.J. and Buchner, F., "Flight Investigation of a Helicopter Low Airspeed Estimation System Based on Measurement of Control Parameters," Proceedings of the Sixth European Rotorcraft and Powered Lift Aircraft Forum, Sept. 1980.

16. Mandle, J., "A Promising Low Speed Air Data System for Helicopters," Proceedings of the Twelfth European Rotorcraft Forum, Sept. 1986.

17. Mueller, B., "An Observer Approach to the Estimation of Helicopter Airspeed," DFVLR-FB-87-13, Braunschweig, April 1987.

AIAA-96-3482-CP

NONLINEAR HEAVE DYNAMICS OF AN AIR CUSHION VEHICLE BAG AND FINGER SKIRT

J. Chung[*] and P. A. Sullivan[**]
University of Toronto, Toronto, Canada
and
T. Ma[***]
Marine Design and Research Institute of China
Shanghai, China

The flexible skirt system used almost exclusively on large amphibious air cushion vehicles such as the U.S. Navy's 160 tonne landing craft is the bag-and-finger skirt, but little information on its dynamical properties is available in the journal literature. The present paper describes results of an analysis of the nonlinear heave dynamics of a simplified configuration chosen specifically to allow formulation from first principles. The skirt mass is lumped in the fingers, with the bag being modelled a combination of massless inelastic membranes and links. Air flow processes are assumed quasisteady, and the bag and cushion volumes are modelled as lumped pneumatic capacitances. The modulation of cushion air escape by skirt-surface contact is also included. The numerical results show that nonlinear effects occur at wave input amplitudes expected to be encountered in practise. At a given frequency, input amplitude increase causes jumps in heave response, period doubling, and chaos. Furthermore, a typical configuration shows a resonance at frequencies at which humans are most sensitive. One solution to this problem may be to reduce skirt mass.

Nomenclature

Roman

A_b = effective area of bag to cushion feed hole orifices, m^2

B_b = width of the vehicle base between inner bag attachment points, m

B_c = width of cushion footprint, m

B_f = finger width, m

c = outer bag chord length

C_b,C_c = pneumatic capacitances of bag and cushion volumes

g = gravitational acceleration, m/s^2

H_c = equilibrium height of craft or model base, m

h_c = height of craft or model base measured upwards from datum, m

h_e = hovergap, or distance between bottom tips of skirt fingers and ground, m

h_f = effective cushion leak height, h_f=$f(h_e,\theta)$, m

h_g = height of ground-plane measured from upwards from datum, m

h_s = skirt height measured downwards from craft base, m

I_s = moment of inertia about center of skirt mass, kg m^2

L_b = length of vehicle base between inner bag attachment points, m

L_c = length of cushion footprint, m

L_M = distance between center of mass of the skirt and inner bag joint, m

L_o = overall length of vehicle including inflated skirt, m

L_{ob} = length of outer bag membrane, m

L_1,L_2, L_3,L_4 = lengths defining finger and inner bag as depicted in Fig. 3., m

L_p = total peripheral length of skirt, m

M_c = total mass of craft including skirt, kg

M_s = mass of skirt, m

P_a = atmosphere pressure, Pa

[*] Research Assistant, Institute for Aerospace Studies,Student member of AIAA.

[**] Professor, Institute for Aerospace Studies.

[***] Senior Engineer, Hovercraft Division.

p_b, p_c = bag and cushion pressure, Pa

Q_a = volume flow from cushion to atmosphere, m^3/s

Q_b = volume flow from fan to bag, m^3/s

Q_c = volume flow from bag to cushion, m^3/s

Q_e = equilibrium flow rate at hover, m^3/s

q_i = generalized coordinates in Lagrangian formulation.

Q_i^{NC} = non-conservative forces acting on skirt due to bag and cushion pressures.

V_b, V_c = Volumes of bag and cushion respectively, m^3

V_i^{ob}, V_i^{ib}, V_i^f = Volume sweeping rate functions for q_i.

Greek

α = angle of link DE relative to horizontal, rad

β = angle of inner bag chord relative to horizontal, rad

γ = angle of link CD relative to horizontal, rad

ν = angle of the inner face of finger relative to horizontal, $\Omega+\gamma$, rad

Ω = angle defining finger geometry, rad

θ = angle of the front face of finger relative to horizontal, rad

ρ = air density, kg/m^3

ω = angular frequency, rad/s

Φ = angle defining finger geometry, rad

Superscripts and Subscripts

a = atmosphere

b = bag

c = cushion

f = finger

ib = inner bag

M = mass center of finger

ob = outer bag

s = skirt

Introduction

Flexible skirted amphibious air cushion vehicles (ACV) are gradually gaining acceptance in certain special purpose marine roles. In North America, the most prominent example is the U.S. Navy's 150 tonne landing craft (LCAC), of which over 76 are now in service. Since 1966 the Canadian Coast-Guard (CCG) has exploited the ACV's amphibious characteristics in search-and-rescue operations in the Vancouver region. More recently, the CCG has used a 37 tonne vehicle in the St.-Lawrence region for ice-breaking and navigational aids servicing; because it has proven more cost-effective in the latter task than a conventional vessel, additional vehicles are being purchased. Vehicles of this type invariably use the bag-and-finger skirt, the principal elements of which are depicted in Fig. 1. Introduced first in Britain in 1966,[1] it has since undergone considerable development. However, it is not free of problems; one is a tendency to produce a rough ride, and another is susceptibility to an instability known as *skirt bounce*. The *antibounce web* in Fig. 1 is used to suppress this instability, but it limits the vertical motion of the skirt, thus interfering with its ability to respond to surface irregularities. In this respect, there is an almost complete absence of investigations of its dynamical properties in the relevant journal literature. The work presented here is a numerical investigation of the nonlinear heave dynamics of this skirt having two objectives. The first is to ascertain if characteristically nonlinear phenomena might be expected to occur in normal craft operation having ride comfort implications, and the second is to give insight into the extent to which linear analysis may be relied upon as a design tool.

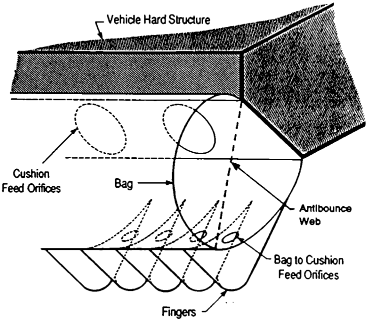


Fig. 1 Main elements of bag and finger skirt.

Analytical and numerical investigations of the simplest possible nonlinear system, a forced linear damped single-degree-of-freedom oscillator with an phenomena including superharmonic and subharmonic resonances, hysteresis, and chaos. All of these are outside the scope of a linear analysis, and all occur in more complex systems. The first phenomenon is a large amplitude response at frequencies which are rational multiples or fractions of the driving frequency. The second occurs when the frequency or amplitude of the input is changed gradually over time. At critical values of either quantity the system responds by abruptly increasing its output amplitude. Then, when the driving frequency or amplitude is decreased slowly, an abrupt decrease in output amplitude occurs at a critical value lower than that at which the increase occurred. Finally, chaos is a non-periodic, apparently random response to a periodic input which usually occurs when a strong nonlinearity exists and when the amplitude of the input is increased sufficiently. The results presented here suggest that all of these phenomena occur for the bag-finger skirt at input amplitudes well within the range likely to be encountered in typical craft operation, and thus may have ride comfort implications.

Now it is a formidable task to develop a realistic model of the dynamics of a flexible-skirted ACV. The geometry of the bag-finger skirt is complex, and can undergo large changes during vehicle motion. Being constructed from elastomer-coated fabrics, the viscoelastic properties of these materials can play a role in skirt dynamics.[2] Furthermore, to adequately model the cushion air escape process, the model must account for intermittent skirt-surface contact around the craft periphery arising from both craft motion and wave action; this can have major effects on the dynamics.[3] It follows that the development of tractable analyses from first principles requires considerable simplication.

The first published analyses used empirical data on skirt deflection.[4-6] In an analysis of a closely related geometry, the bag-pericell skirt, Carrier et al [7] assumed the bag to be rigid. To allow consideration of bag dynamics, Ma and Sullivan investigated a two-dimensional section of the skirt constrained to move only in heave and subject to pure heave or long-wave inputs.[8] They formulated a multiple-degree-of-freedom model using a Lagrangian technique to account for the inputs to the fluid mechanics from bag and cushion volume changes. Implementation of this model requires making assumptions about the skirt motion, and Sullivan, Charest and Ma[9] applied this formalism to predict the heave stiffness of a small laboratory model constructed to conform with the theory. They modelled the fingers as rigid bodies, and the bag as combinations of massless inelastic membranes. The predictions agreed closely with experiment, including showing the effects of skirt geometry changes.

The present work extends the analysis of Sullivan, Charest and Ma[9] to investigate the heave dynamics. The nonlinear dynamics equations for their model are formulated and solved numerically to obtain frequency response characteristics. The code is validated by comparison of its static predictions with the experiments of Sullivan,Charest and Ma[9], and by comparison of its frequency response predictions with a separate linear analysis. Effects of input amplitude on frequency response are presented for a configuration representative of the CCG vehicle noted above, and for conditions representative of those encountered by such a vehicle in typical operations.

The model of Sullivan and Ma[8] does not include an antibounce web, since the aim is to explore the dynamics of a system having the maximum possible degree of responsiveness to surface irregularities. Also, to distinguish the effects of skirt motion from other cushion phenomena, the results are compared with those obtained by preventing bag deformation. The latter *rigid skirt* results also provide insight into the

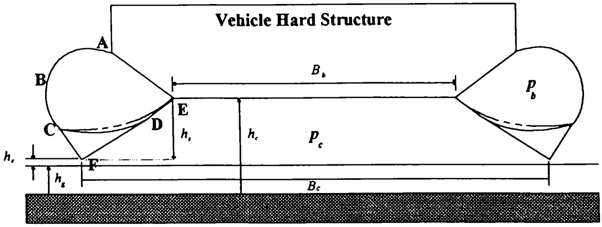


Fig. 2 Configuration investigated in Refs. 8 and 9.

latter *rigid skirt* results also provide insight into the effect of an antibounce web since, as noted above, the main effect of such a device appears to be limitation of vertical skirt motion.

Assumptions used in Analysis

The assumptions used here fall into two groups: those relating to the skirt structure, and those relating to the fluid mechanics. Consider these in turn.

Skirt Structure

Figure 3 depicts the main elements of the skirt model; it comprises three distinct parts. The first is the *outer bag*, which is the part ABC of the skirt between upper attachment point A and the outer edge of the fingers, C. The second part is the *inner bag* CDE, and the third is the *fingers* CDF. The outer bag is assumed to be a massless inelastic membrane subject to a spatially uniform bag pressure p_b, so that it is always a circular arc. The inner bag is assumed to comprise two rigid massless links having lengths L_1 and L_2 respectively. With p_c being the assumed spatially uniform presssure in the cushion volume V_c, these links are subject to the pressure difference p_b-p_c. The finger is a rigid body having mass and moment of inertia attached to the link CD; the part CF is subject to p_c. The skirt is has its mass concentrated at the centroid of the finger, and has two degrees-of-freedom corresponding to the angles α and γ defined in Fig.3.

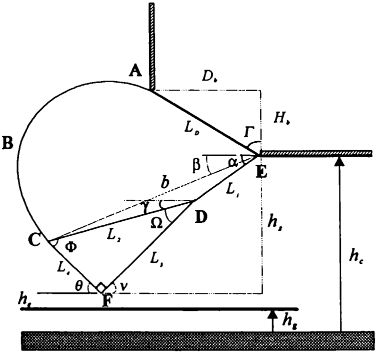


Fig. 3 Elements of skirt model used in present analysis.

The above model is called the *two link* model to distinguish it frm the *two membrane* model used by Ma and Sullivan.[8] Depicted in Fig. 2, the inner bag is assumed to be an inelastic circular arc membrane. The finger is attached to the bag at the point C and to the craft base at E by a rigid link in such way that it does not touch the underside of the inner bag. It was suggested that these two models encompass the behavior of the skirt in practise. Sullivan, Charest and Ma[9] showed that, at least as far as static response is concerned, there is little difference between the two models.

As depicted in Fig.2, the vehicle or *hard structure* dynamics is described by the height $h_c(t)$ of the vehicle base above a datum, and the surface input by a specified ground height $h_g(t)$. Then, with h_s being the skirt height and h_e being the hovergap, or distance between the bottom tips of the fingers and the ground,

$$h_g + h_e + h_s = h_c \qquad (1)$$

where, from Fig.3, the skirt height is given by

$$h_s = L_1 \sin\alpha + L_2 \sin(\Omega + \gamma) \qquad (2)$$

The analysis below also allows for the possibility that the skirt contacts the ground, corresponding to $h_e < 0$. Consistent with observations of such systems, when this occurs, it is assumed that skirt collapse occurs only locally, with the finger geometry being that above the intersection of the uncollapsed finger and the ground-plane.

Fluid Mechanics

The air flow processes from the vehicle lift fan into the bag (Q_b), from bag to cushion through the orifices depicted in Fig.1 (Q_c), and from cushion to atmosphere through the hovergap h_e depicted in Fig.3 (Q_a) are all assumed to be quasisteady, with the latter two described by orifice-flow laws based on Bernoulli's law,[3] and suitable discharge coefficients. For the flow through h_e, the discharge coefficients depends on the finger geometry through both h_e and the finger orientation angle θ. The volumes V_b and V_c are modeled as lumped pneumatic capacitances, this being included because the compressibility of the cushion air has been shown to significantly affect dynamics under certain operating conditions, and can be a source of dynamic instability.[10]

The fluid dynamics is coupled to the skirt and craft dynamics through mass conservation laws for the variable V_b and V_c, and through the modulation of h_e.

Finally, we note that, with the possible exception of the lift fan response to fluctuating p_b,[11] it has been shown that these assumptions provide a good model of the fluid dynamics.[3]

Equations of Motion

The equations of the craft dynamics, including that of the skirt, are formulated using the method of Lagrange. With T and P being, respectively, the system kinetic and potential energies, Lagrange's equations are:

$$\frac{d}{dt}\left(\frac{\partial T}{\partial \dot{q}_i}\right)-\frac{\partial T}{\partial q_i}+\frac{\partial P}{\partial q_i}=Q_i^{NC},\ \text{i=1,2,3}$$

where $q_1 = h_c$, $q_2 = \alpha$,and $q_3 = \gamma$. The non-conservative forces Q_i^{NC} arise from the pressure forces acting in V_b and V_c.

The Q_i^{NC} are calculated by considering the work done hydrostatically by p_b acting on the outer bag, by p_b-p_c acting on the inner bag, and p_c acting on the fingers. In Ref.8 it is shown that they take the form

$$Q_i^{NC}=p_b V_i^{ob}+(p_b-p_c)V_i^{ib}+p_c V_i^{f}$$

where V_i^{ob} , V_i^{ib} and V_i^{f} are the rates, with respect to the q_i , at which volumes are swept out by the outer bag, inner bag and finger volumes.

For the above system,

$$\begin{aligned}T=&\frac{1}{2}M_c\dot{h}_c^2+\frac{1}{2}I_s\dot{\gamma}^2\\&+\frac{1}{2}M_s[L_1^2\dot{\alpha}^2+L_M^2\dot{\gamma}^2+2L_1L_M\cos(\gamma_M-\alpha)\dot{\alpha}\dot{\gamma}\\&-2\dot{h}_c(L_1\cos\alpha\dot{\alpha}+L_M\cos\gamma_M\dot{\gamma})]\end{aligned}$$

$$P=M_c g h_c-M_s g(L_1\sin\alpha+L_M\sin\gamma_M)$$

Hence the three equations of motion are:

(A) $q_1 = h_c$

$$\begin{aligned}&M_c\ddot{h}_c+M_s[L_1(\sin\alpha\dot{\alpha}^2-\cos\alpha\ddot{\alpha})\\&+L_M(\sin\gamma_M\dot{\gamma}^2-\cos\gamma_M\ddot{\gamma})]+M_c g=p_c\frac{\partial V_c}{\partial h_c}\end{aligned} \quad (3)$$

(B) $q_2 = \alpha$

$$\begin{aligned}&M_sL_1[L_1\ddot{\alpha}-L_M\sin(\gamma_M-\alpha)\dot{\gamma}^2+\\&L_M\cos(\gamma_M-\alpha)\ddot{\gamma}-\ddot{h}_c\cos\alpha-g\cos\alpha]\\&=p_bV_\alpha^{ob}+(p_b-p_c)V_\alpha^{ib}+p_cV_\alpha^{f}\end{aligned} \quad (4)$$

(C) $q_3 = \gamma$

$$\begin{aligned}&M_sL_M[L_M\ddot{\gamma}+L_1\sin(\gamma_M-\alpha)\dot{\alpha}^2+\\&L_1\cos(\gamma_M-\alpha)\ddot{\alpha}-\ddot{h}_c\cos\gamma_M-g\cos\gamma_M]+I_s\ddot{\gamma}\\&=p_bV_\gamma^{ob}+(p_b-p_c)V_\gamma^{ib}+p_cV_\gamma^{f}\end{aligned} \quad (5)$$

The functions V_i^{ob},V_i^{ib} and V_i^{f} are given in the Appendix.

Air Flow Equations

The vehicle lift fans are assumed to deliver air to the bag according to the law

$$\frac{p_b}{p_{br}}=\xi-(\xi-1)(\frac{Q_b}{Q_{br}})^3 \quad (6)$$

In this expression p_{br} and Q_{br} are, respectively, a reference pressure and flow; they are usually taken to be those at the craft's design operating equilibrium conditions. The quantity ξ is the pressure ratio corresponding to Q_b= 0; usually $1.2 \leq \xi \leq 1.4$ to ensure adequate stability margin. The form of equation (6) is chosen to reflect the fact that periodic flow reversal can occur,[12] and that for $Q_b < 0$, $dp_b/dQ_b < 0$.[1]

The flow from bag to cushion is described by

$$Q_c=A_b\,\text{sgn}(p_b-p_c)\sqrt{\frac{2|p_b-p_c|}{\rho}} \quad (7)$$

where A_b is the effective area of the orifice separating the bag and cushion. This form also allows for flow reversal through the bag orifices to occur. Even in the presence of flow reversal, the model of eqn. (7) is usually adequate.[3]

The flow escaping from the cushion to atmosphere is modelled by

$$Q_a=L_p h_f(h_e,\theta)\,\text{sgn}(p_c)\sqrt{\frac{2|p_c|}{\rho}} \quad (8)$$

where L_p is the total peripheral length of the skirt and h_f is an effective *leak height*, which is a function of both h_e and the external slope of the finger relative to the ground. With $\theta = \Phi - \gamma$, where Φ is the finger geometry angle depicted in Fig.3, h_f is expressed in the form

$$h_f=B_f f(\frac{h_e}{B_f},\theta) \quad (9)$$

Figure 4 gives typical computed and measured values of h_f/B_f obtained from a small model.[9] The computation assumes an area available for flow equal to the area between fingers and ground projected onto the vertical plane, together with a discharge coefficient dependent on θ. The difference between computation and experiment is caused by leaks at the end-plates of the model; this is not present in a full-scale vehicle. Figure 4 shows that, as h_e decreases from $h_e < 0$, h_f initially depends linearly on h_e. Then, when surface contact occurs at $h_e = 0$, further decrease in h_e causes h_f to decrease nonlinearly as the circular-arc tips of the fingers (at F in Fig. 3) collapse, shutting off the flow.

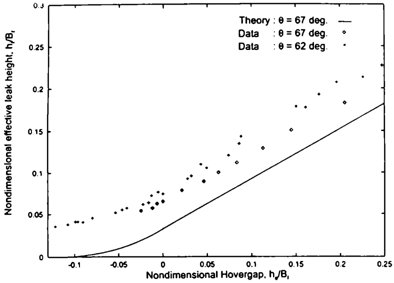


Fig. 4 Predicted and measured values of h_f as a function of h_e.

Finally, as $p_b/P_a \rightarrow 0$ and $p_c/P_a \rightarrow 0$, conservation of mass for the bag and cushion volumes take the form

$$C_b \dot{p}_b + \dot{V}_b = Q_b - Q_c \tag{10}$$

$$C_c \dot{p}_c + \dot{V}_c = Q_c - Q_a \tag{11}$$

In these equations, Q_b and Q_c are the respective pneumatic capacitances of V_b and V_c, given respectively by $C_b = V_b/nP_a$ and $C_c = V_c/nP_a$. Here n is the polytropic exponent in the law $(P_a + p)/\rho^n =$ constant; for air cushion dynamics, we can take n equal to the ratio of specific heats,[10] which is 1.4 for air. Also, for the geometry of Fig. 3., $V_b = V_b(\alpha, \gamma)$ and $V_c = V_c(\alpha, \gamma, h_e)$; the expressions for these quantities are given in the Appendix.

Numerical solution

Numerical solutions to equations (1)-(11) were obtained for surface inputs of the form $h_g(t) = h_{gl}\sin(2\pi f t)$ where $f = \omega/2\pi$ is the input frequency. Both the craft parameters and values of f and h_{gl} were selected as described below. Equations (1)-(11) constitute an eighth-order system which, owing to the pneumatic capacitances, is very stiff. Satisfactory solutions are obtained by using a polynomial extrapolation technique due to Gear.[13] Initial or equilibrium hover conditions are specified by solving equations (1)-(11) with all time derivatives set to zero, and with $h_g = 0$. The rigid skirt results are obtained by fixing α and γ at their equilibrium hover values, α_0 and γ_0 ; in this case the system of equations is fourth order. The program was validated in two ways: by comparison of its static heave stiffness predictions with measurements obtained from a small laboratory model, and then by comparison of the frequency response characteristics for small h_{gl} with those obtained from a completely separate linear analysis. Figure 5 plots computed and measured M_c as a function of H_c for the laboratory model used by Sullivan, Charest and Ma. It has the parameters given in Table 1. The agreement between prediction and measurement is virtually identical to that obtained by Sullivan, Charest and Ma,[9] who solved the static equations directly using a different computer programme. The results of comparisons with the linear predictions are described below.

To establish the limits of linear response, for a given ω, Fourier analysis is applied to the craft heave histories $h_c(t)$ corresponding to a range of h_{gl}. This gives

$$h_c(t) = h_{c0} + h_{cl}\sin(\omega t + \phi) + \ldots \tag{12}$$

For linear response, h_{c0}, the ratio h_{cl}/h_{gl}, and ϕ are all independent of h_{gl}. Departure from this behavior is used as an indicator of nonlinearity.

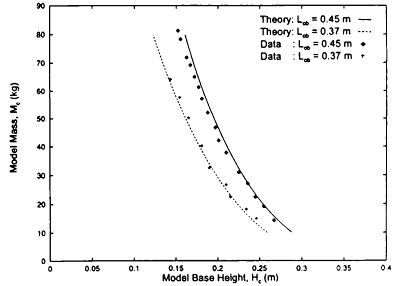


Fig. 5 Predictions and measurements of static stiffness for the model of Ref.9 and for two skirt geometries.

Table 1. Configuration used in simulations

	Craft				Skirt						Lift Fan			
Case	B_b (m)	L_b (m)	L_c^* (m)	M_c (kg)	L_1 (m)	L_2 (m)	L_3 (m)	L_4 (m)	L_{ob} (m)	Ω (rad)	A_b (m^2)	p_{br} (Pa)	Q_{br} (m^3/s)	ξ
Laboratory model	0.572	0.914	0.914	44	0.048	0.362	0.3	0.2	0.37	0.585	0.027	800	0.59	50
37 tone craft	5.75	18.24	21	36,741	0.18	2.02	1.69	1.1	2.6	0.576	3.695	2470	95.78	1.2

Note : * For model $L_b = L_c$, and ξ = 50 approximates the constant Q_b used in the experiments.

Table 2. Equilibrium conditions

	Flow				Skirt				
Case	p_c (Pa)	Q_e (m^3/s)	p_b/p_c	h_e^* (m)	α_0 (rad)	γ_0 (rad)	H_c (m)	h_{f0} (m)	h_{s0} (m)
Laboratory Model	411.3	0.59	1.7	0.009	0.591	0.012	0.205	0.012	0.195
37 tonne craft	2,016.5	96.8	1.2	0.004	0.501	0.189	1.258	0.028	1.254

Note : * For the model, the fingers were clear of the ground, whereas for the craft substantial fingers contact

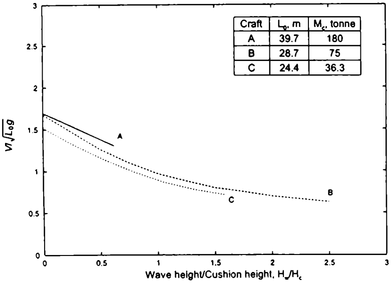


Fig. 6 Manufacturer's data for craft speed in beam seas as a function of wave height.

Parameters used in the Calculations

Table 1 gives the parameters used to obtain the results presented below. These parameters are selected to be representative of the 37 tonne CCG vehicle, "Waban-Aki".[14] Since manufacturer's data on most of the items in Table 1 are not published, these were estimated as described below.

The equilibrium p_c was estimated from M_c, and from the cushion footprint length L_c and width B_c. Then with $S_c = L_c B_c$ being the cushion footprint area, the value of Q_e was estimated from

$$\frac{Q_e}{\sqrt{\dfrac{2S_c^2 p_c}{\rho_a}}} = 4.5\times10^{-3} + 5.9\times10^{-4}\left(\frac{p_c}{\rho_a g L_c}\right), \quad (13)$$

which correlates operating experience of vehicles built in China. For Waban-Aki, this gives h_f = 28.4 mm; observation of the finger geometry of the vehicle when hovering on a smooth surface and the data of Fig. 4 suggests that this estimate of Q_e is appropriate. A value of p_b/p_{ce} = 1.2 was used to calculate p_{be}; this is consistent with operating data for Waban-Aki. Equation (7) and the above data was then used to calculate A_b. Finally, measurements of the vehicle's overall skirt geometry at hover, and of the weight of a sample of replacement skirt material were used to estimate skirt parameters.

To calculate representative frequencies and maximum input amplitudes, the following procedure was adopted. Figure 6 gives data taken from the manufacturer's sales brochures for craft operational speed V_c as a function of encountered trough-to-crest wave height H_w. To allow comparison of this data for craft of different size, with L_0 being the overall craft length, Fig. 6 plots $V_c/\{L_0 g\}^{1/2}$ against H_W/H_C. This data is for beam seas. To estimate an equivalent input frequency corresponding to a maximum $h_{gl} = H_W/2$ for the head seas case, V_c is assumed to be 2/3 of that given by Fig. 6. Then f is estimated from the expression

$$f = \sqrt{\frac{g}{2\pi R H_W}} + \frac{V_c}{R H_W} \qquad (14)$$

In this expression the first term is the contribution due to the speed of the approaching wave as calculated by the formula for small amplitude waves, and the second is due to the motion of the craft. The quantity RH_W is the estimated length of the approaching wave; we assume $R = 10$.

Results

Small Input Frequency Response

Figures 7 and 8 compare amplitude and phase responses with the linear predictions for the two-link skirt model for M_s = 900 kg, and Fig. 9 gives the amplitude response for the rigid skirt case. For h_{gl} = 0.1 mm, the linear and nonlinear results are indistinguishable and, for h_{gl} = 1.0 mm, observable nonlinear effects occur only in the neighborhood of the second resonance peak at f = 3.94 Hz. For the rigid skirt case, the results for h_{gl} = 1.0 mm. are indistinguishable from the linear response, but at h_{gl} = 5.0 mm. significant deviation occurs. The deviation from linear response in the neighborhood of the 3.94 Hz. resonance in Figs. 7 and 8 is associated with the very large motion of the skirt interacting with the flow function nonlinearity depicted in Fig. 4.

Considering the features of the response, Figs 7-9 show that skirt flexibility substantially reduces cushion heave stiffness, so that the resonance peak associated with craft mass moves from 2.62 Hz to 0.76 Hz. However, a large resonance peak associated with the skirt mass M_s now appears at 3.94 Hz. Clearly this is undesirable, since humans tend to be most sensitive to this type of motion in the range 4-8 Hz.[15] The response obtained by reducing M_s from 900 kg to 180 kg is also plotted in Figs 7 and 8; this eliminates the problem.

Effect of Input Amplitude on Response

Figures 10 and 11 give the effect at ω = 15 rad/s on $h_c(t)$ of slowly increasing and then decreasing h_{gl} in the range 0.01 ≤ h_{gl} ≤ 0.32 m. This value of ω

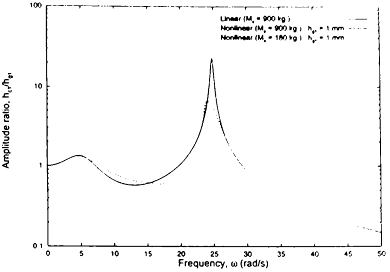


Fig. 7 Heave amplitude response as predicted by linear analysis and nonlinear simulation for the two-link model.

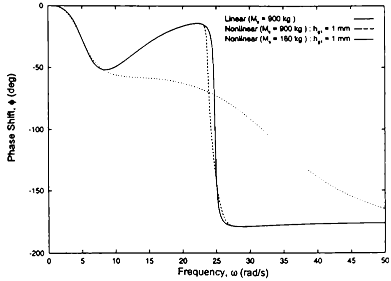


Fig. 8 Phase response as predicted by linear analysis and nonlinear simulation for the two-link model.

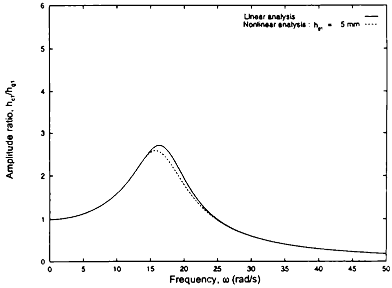


Fig. 9 Heave amplitude response as predicted by linear analysis and nonlinear simulation for the rigid model.

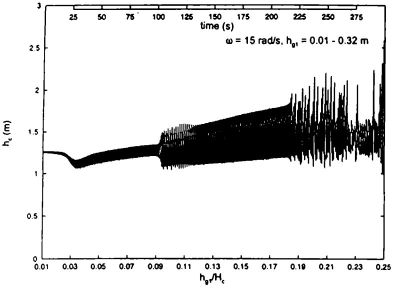


Fig. 10 History of craft response to a slowly increasing ground input amplitude.

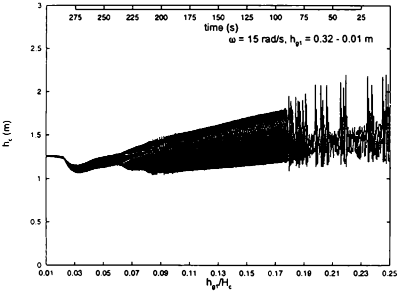


Fig. 11 History of craft response to a slowly decreasing ground input ampltude.

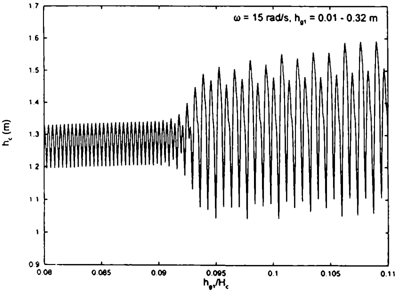


Fig. 12 Details of period doubling in Fig. 10.

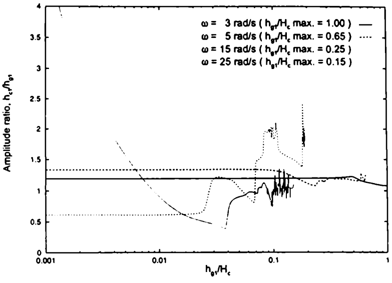


Fig. 13 Effect of input amplitude on the response amplitude ratio for four ω.

was chosen to be high enough to involve ride comfort implications while avoiding the skirt mass resonance. Also, eqn. (14) gives a peak h_{gl} = 0.32 m. at ω = 15 rad/s. To obtain these results, the duration of the interval over which the change in h_{gl} was imposed was increased until, when plotted against the linearly changing h_{gl}, the envelopes of the h_c histories were not affected.

The graphs demonstrate essentially linear behavior for $0.010 \leq h_{gl}/H_c \leq 0.022$; that is, h_{c0} remains constant and equal to H_c, and h_{cl} is very nearly proportional to h_{gl}. As h_{gl} increases, the first significant nonlinear effect is an abrupt decrease in h_{c0}, followed by a period doubling at $h_{gl}/H_c = 0.091$, which abruptly increases the amplitude of the craft heave motion by a factor in excess of three. Figure 12 gives the details of the transition. At $h_{gl}/H_c = 0.18$, $h_c(t)$ ceases to be periodic and the amplitude of the motion becomes erratic. Both phase-plane plots and Poincarѐ maps of $\dot{h}_c$ show that this motion exhibits the phenomena characteristic of chaos.[16] Finally, as h_{gl} decreases, chaotic motion abruptly ceases at about the value of h_{gl}/H_c at which it first occurred, but period-doubling dies out only when h_{gl}/H_c decreases to 0.061, a value significantly lower than at onset.

Figure 13 plots a measure of the ratio of the amplitude of craft oscillations in $h_c(t)$ to that of $h_g(t)$, namely h_{gl}, against h_{gl}/H_c. Prior to period-doubling, the measure of the oscillation amplitude is the amplitude of the fundamental of $h_c(t)$ as determined by Fourier analysis. At the onset of period-doubling the amplitude is taken to be one half of the peak-to-peak motion. In each case the graph is terminated at the maximum value of h_{gl}/H_c. This suggests that, except in the neighborhood of the resonance at ω = 25 rad/s., prior to the onset of period-doubling, the craft response is predominantly linear.

Discussion and Conclusion

The present results indicate that characteristically nonlinear dynamical phenomena such as period-doubling and chaos can be expected to occur during the normal operation of air cushion vehicles equipped with the bag-and-finger skirt. But an important feature, revealed also by the linear analysis, is the presence of a resonance associated with skirt mass at a frequency which, for the 37 tonne vehicle investigated, is close to the range at which humans are most sensitive.

Initially those involved in the development of air cushion vehicle skirts were divided into two schools of thought: one advocating very light skirts to obtain responsiveness, and the other advocating heavier skirts for practical reasons such as resistance to wear and damage. The latter school has prevailed for vehicles using the bag-and-finger skirt, with an elastomer-coated fabric such as neoprene-nylon being the material of choice. The present results suggest that skirt response might be improved by using a modern high-strength material to reduce mass except for those parts where wear due to surface contact may be expected.

Acknowledgement

The present research was supported in part by the Natural Sciences and Engineering Research Council of Canada.

References

[1]Mantle, P. J., Air Cushion Craft Development, U.S. Navy, David W. Taylor Naval Ship Research and Development Center, Bethesda, Md., DTNSRDC- 80/012 Jan. 1980.

[2]Graham, T. A., and Sullivan, P. A., and Hinchey, M. J., "Skirt Material Effect on Air Cushion Dynamic Heave Stability", *Journal of Aircraft*, Vol.22, No.2, Feb. 1985, pp. 105-108.

[3]Sullivan, P. A., Byrne, J. E. , Hinchey, M. J., "Nonlinear Oscillations of a Simple Flexible Skirt Air Cushion", *Journal of Sound and Vibration*, Vol. 102, Oct. 1985, pp. 269 - 283.

[4]Reynolds, A. J., West, R. P., and Brooks, B E., "Heaving and Pitching Response of a Hovercraft Moving Over Regular waves", *Journal of Mechanical Engineering Science*, Vol. 14, Oct. 1972, pp. 340-352.

[5]Reynolds, A. J., "A Nonlinear Theory for a Hovercraft Moving over Regular Waves", *Journal of Mechanical Engineering Science*, Vol. 16, No. 5, 1974, pp. 310-316.

[6]Reynolds, A. J., and Brooks, B. E., "The Roles of Skirt Planform and Deflection in Hovercraft Response", *Journal of Mechanical Engineering Science*, Vol. 19, No.1, pp. 1-12

[7]Carrier, R., Magnuson, A. H., and Swift, M. R., "Seakeeping Dynamics of a Single Cushion Peripheral Cell-Stabilized Air Cushion Vehicle", *Journal of Hydronautics*, Vol. 12, No. 2, 1978, pp. 49-54.

[8]Ma, T. and Sullivan, P. A., "Dynamics of Responsive Skirts", Hovercraft Technology, Economics and Applications, J. R. Amyot, ed., Elsevier Studies in Mechanical Engineering, Vol.11,Elsevier, Amsterdam, 1989,pp.526-600.

[9]Sullivan,P.A., Charest, P.A. and Ma, T., "Heave Stiffness of an Air Cushion Vehicle Bag and Finger Skirt", *Journal of Ship Research*, Vol. 38, No. 4, Dec. 1994, pp.302-307

[10]Sullivan, P. A., Hinchey, M. J., and Green, G. M., "A Review and Assessment of methods for the Prediction of the Dynamic Stability of Air Cushions", *Journal of Sound and Vibration*, Vol. 54, No. 3,Oct. 1982.

[11]Sullivan, P. A., Gosselin, F., and Hinchey, M. J.,"Dynamic Response of an Air Cushion Lift Fan", *Proceedings of the 1992 Intersociety High Performance Marine Vehicles Conference*, June 24-27, 1992 (American Society of Naval Engineers),pp. ACV39-47.

[12]Graham,T.A.,and Sullivan, P. A.,"Pitch-Heave Dynamics of a Segmented Skirt Cushion", *Proceedings of the 1989 Intersociety Advanced Marine Vehicle Conference*, AIAA, Arlington, Va,pp.334-347.

[13]Gear, G. W.,"Numerical Initial Value Problems in Ordinally Differential Equations", Prentice Hall, Englewood cliffs, New Jersey,1971.

[14]D L'Heureaux, Canadian Coast Guard, private communication.

[15]"Ride and Vibration Data Manual",SAEJ6a, Society of Automotive Engineers, Warrendale, PA,December 1965.

[16]Moon, F. C., *"Chaotic Vibrations"*, John Wiley & Sons 1987.

APPENDIX

The quantitiies V_i^{ob},V_i^{ib} and V_i^f in eqns 3-5 are expressed below per unit length of skirt; that is, for example $V_i^{ob} = L_p\bar{V}_i^{ob}$. Also the quantities b, β, L_D and Γ are defined in Fig.3.

For the outer bag of length L_{ob}, the chord length c, and arc semi-angle σ_0 are given respectively by

$$c^2 = L_D^2 + b^2 + 2L_D b \sin(\beta - \Gamma)$$

$$\frac{\sin\sigma_0}{\sigma_0} = \frac{c}{L_{ob}}$$

so that

$$\bar{V}_\beta^{ob} = -b[b + L_D\{\sin(\beta - \Gamma) + \cot\sigma_0 \cos(\beta - \Gamma)\}]/2$$

and

$$\bar{V}_b^{ob} = [L_D \cos(\beta - \Gamma) + \cot\sigma_0\{b + L_D \sin(\beta - \Gamma)\}]/2$$

where, from Fig.3,

$$b^2 = L_1^2 + L_2^2 + 2L_1L_2\cos(\alpha - \gamma)$$

$$\tan\beta = \frac{L_1\sin\alpha + L_2\sin\gamma}{L_1\cos\alpha + L_2\cos\gamma}$$

Hence

$$\bar{V}_i^{ob} = \bar{V}_\beta^{ob}\frac{\partial\beta}{\partial q_i} + \bar{V}_b^{ob}\frac{\partial b}{\partial q_i}$$

for q_2 = α and q_3 = γ.

For the inner bag

$$\bar{V}_\alpha^{ib} = \frac{(L_1 + L_2)^2}{2}; \bar{V}_\gamma^{ib} = \frac{L_2^2}{2}$$

For the $\bar{V}_i^f$, and for $\partial V_c/\partial h_c$ define a skirt wetted or ground penetration height h_w as

$$\begin{aligned} h_w &= h_s - h_c \quad \text{if } h_c < h_s \\ &= 0 \qquad\quad\ \text{if } h_c \geq h_s. \end{aligned}$$

Then the fraction of that part of the finger CF from Fig.3 subject to P_c is taken as

$$l_4 = L_4 - h_w \operatorname{cosec}(\Phi - \gamma),$$

so that

$$\bar{V}_\alpha^f = L_1 l_4 \cos(\alpha + \Phi - \gamma),$$

$$\bar{V}_\gamma^f = l_4[L_2\cos\Phi - \frac{l_4}{2}]$$

Then $\partial V_c/\partial h_c$ is the foot print area of the cushion, given by

$$\begin{aligned} \frac{\partial V_c}{\partial h_c} = {} & L_b B_b + \\ & L_p\{L_1\cos\alpha + L_3\cos(\Omega + \gamma) + h_w\cot(\Phi - \gamma)\} \end{aligned}$$

Finally

$$\begin{aligned} V_b = L_p[& \frac{L_{ob}^2}{8}\{\frac{2\sigma_0 - \sin 2\sigma_0}{\sigma_0^2}\} + \\ & \frac{D_B b\sin\beta + H_B b\cos\beta}{2} + \frac{1}{2}bL_1\sin(\alpha - \beta)] \end{aligned}$$

and

$$\begin{aligned} V_c = {} & B_b L_b(L_1\sin\alpha + L_3\sin\nu + h_e) + \\ & L_p[\frac{1}{2}L_1\cos\alpha(L_1\sin\alpha + 2L_3\sin\nu) + h_e L_1\cos\alpha + \\ & \frac{1}{4}L_3^2\sin 2\nu + h_e(L_3 - l_3)\cos\nu - \frac{h_w^2}{2}\cot\nu + \\ & \frac{1}{2}L_2L_3\sin\Omega - \frac{h_w^2}{\sin 2\nu}] \end{aligned}$$

where $l_3 = h_w/\sin\nu$ is the submerged part of L_3.

AIAA-96-3483-CP

REAL-TIME SIMULATION OF RADIO-CONTROLLED HELICOPTERS

Martin G Kellett[1]
Royal Military College of Science (Cranfield University)
Shrivenham, SWINDON, SN6 8LA, ENGLAND

ABSTRACT

The sport of flying radio-controlled model helicopters has expanded rapidly in recent years, mostly driven by the challenge of learning to control a most difficult vehicle.

This paper describes HELIX, *a radio-controlled simulation of model helicopters which is intended to serve as a training aid for models pilots. The simulation runs in on a PC-compatible, providing the pilot with a real-time 3D display. The system is controlled by the pilot's own radio transmitter, which interfaces to the computer through a custom-designed FM receiver.*

An overview of the system is presented, and the flight equations used are discussed, particular attention being paid to the difficulties associated with realtime implementation on a computer of limited computational ability.

INTRODUCTION

The leisure activity of flying radio-controlled aircraft is well established, but it is only in recent years that practical flying model helicopters have become available. For the pilot, all the difficulties of flying full-size helicopters are present in models, together with additional problems due to the lack of "seat-of-the-pants" motion sensing, and the 3D axis transformations necessary due to the changing attitude of the helicopter relative to the pilot.

These difficulties result in a large number of model helicopter crashes which are solely attributable to pilot error, usually through mistakes in mentally tracking the orientation of the vehicle, or simply moving a control stick the wrong way. The recent increases in personal computing power have now enabled the development of simulation training aids for model helicopter pilots which allow practising of these disorienting manoeuvres in safe environment. The main objective of simulation is to train the pilot responses and reactions, in much the same way that pilots of full-size aircraft are closely supervised and guided through a carefully design flying syllabus. Model pilots are inevitably flying solo from day one however, and are not constrianed by any prescribed training programme.

The requirements for any flying training simulator, either model or full-size, are well established. Display quality and refresh rate, together with accurate dynamic responses being most important. A requirement less frequently considered is that a simulation must not provide any "negative" training, e.g. require training to operate the simulator which does not apply to the real aircraft

This paper describes HELIX, a real-time simulation of radio controlled helicopters and fixed-wing aircraft. The system is a software package for PC compatibles, together with a FM receiver interface which allows the pilot to make use of an existing radio control transmitter. This feature is important largely because of the negative training requirement. Model pilots operate their controls without any visual reference and become very familiar with the characteristics of their own equipment. Providing an alternative control box would be a major distraction for the trainee, who might then be uncomfortable making the transition to the real model. This aspect of simulation is common in full-size systems, where great effort is made to duplicate the cockpit and control configuration of the real aircraft.

[1] Lecturer, Department of Aerospace & Guidance Systems, Member AIAA

System Overview

Hardware

Three components make up the hardware configuration for a HELIX system. The host computer runs the simulation software and provides a visual display, the radio transmitter is the control box used by the pilot, and a FM receiver interface allows the computer to receive data transmitted by the controller.

HOST COMPUTER. For practical commercial reasons, HELIX is designed to run on any IBM-compatible PC. Minimum requirements are a 50 MHz 486 processor, with numeric coprocessor due to the high computational demands, a hard disk, 4 Mb RAM and a standard VGA graphics card.

Until recently this was quite a demanding specification, but modern entry-level machines usually exceed these requirements by a large degree. This highlights one of the important aspects of choosing a PC for the host. In addition to having the largest market penetration, the demand for backward compatibility ensures that all future machines will run the software, and will attain higher and higher performance without any software alterations. Equally, this allows further software development with more sophisticated flight models and visual detail to follow the trend of increasing computational ability.

TRANSMITTER. Radio control transmitters are small hand-held units containing a pair of two axis joysticks, with additional switches and controls for trimming and changing of operating modes. Modern units are very complex devices, and usually incoporate a LCD panel for programming the response characteristics. The details vary considerably from unit to unit, and learning how to program the transmitter effectively is an important use for the simulator. For this reason, HELIX is designed to operate exclusively with the user's own transmitter.

RECEIVER. In order to allow the PC to access the commands from the transmitter, a FM radio receiver interface has been developd which converts the pulse position modulated signals (PPM) into RS-232 data which the PC receives through a COM port. The reciever is a special low-power design, and can thus derive its power supply from two of the COM port handshake lines, which can supply up to 10 mA. Versions are available for 72 MHz use in the US, and the lower frequency bands of 35 and 40 MHz used in Europe. For circumstances where it is inappropriate to use FM radio transmissions, the receiver also has a socket to allow direct connection to the transmitter, taking advantage of the "buddy-box" facility provided by most manufacturers.

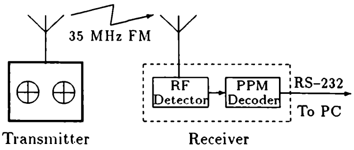


Figure 1: HELIX Transmitter & Receiver

Software

The software for HELIX is mostly written in Borland C, but all the time-critical sections such as graphics and serial data input are written directly in 80x86 assembly language. The real-time nature the simulation dictates that an interrupt-driven scheme be adopted to desynchronise the graphics and flight model calculations. The flight model is executed at a constant rate (36.4 Hz) as a background task, while the graphics runs as a foreground program using data from the latest valid iteration of the flight model. This allows the graphics refresh rate to gracefully degrade on machines with limited computational capacity, without the dynamics of the simulation being compromised.

Serial data from the receiver is managed under interrupt, with an appropriate service routine assembling the packets of data before being made available to the flight model.

OPERATING FEATURES. The program has a mouse-driven menu system to allow the user to select options and configurations, and a context-sensitive help system provides additional information when required. Options allow the user to select the model to be simulated, scenery to be used, wind and turbulence levels etc.

Further options allow the user to program the control characteristics in the same manner as a programmable transmitter. This allows users with more basic equipment to experiment with the response shapes and mixings available with more expensive radio gear.

Another menu allows access to previous flight recordings for analysis. The data from every flight is automatically recorded internally by HELIX, and can be saved to disk if required. Previous recordings recalled and replayed, with fast forward, rewind and pause options.

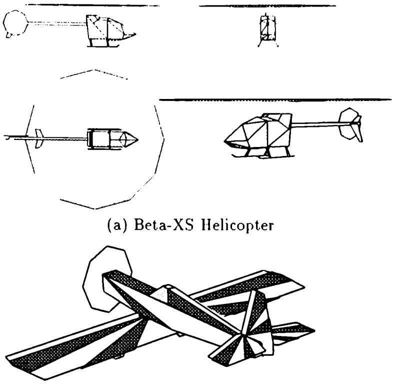
VISUAL DISPLAY. To be an effective aid, a simulation must reproduce the cues used by a model pilot, and the most important of these is the visual display. This need not be highly detailed, but should provide enough visual information for the pilot to accurately determine the position and rates of the vehicle, in both translational and rotational senses. The refresh rate of the display must be fast enough to avoid any perception of delay, and this factor is found to be more important than providing high levels of display detail.

If computing capacity permits, HELIX will refresh the display at the same 36.4 Hz rate as the flight equations are calculated. If computing capacity is limited (or scene content is too high), the graphics system will simply draw new frames as fast as it can, leaving the flight model unchanged at 36.4 Hz. Three different levels of scene detail are user-selectable for such circumstances.

The display in HELIX is very simple, using one of the 16-colour VGA modes at 640×350 pixel resolution. The models are rendered using polygons, typically using about 100 for each vehicle (figure 2). Scenery consists of flat shading for earth and sky, simple near and far field objects (hills, trees, cones etc), ground markings and a dot pattern on the ground surrounding the pilot to provide basic texture cues.

The viewpoint of the display is that from a pilot in a fixed position. The display is automatically rotated in azimuth and elevation to track the moving aircraft. The camera motion is smoothed in eahc axis through a first order lag, which can have time constants modified by the user to provide fast, medium or slow tracking. An alternative camera mode simulates a chase aircraft and will follow the model around the sky, attempting to remain a fixed distance from it. The motion of this chase mode is again smoothed to provide a mor realistic display.

AUDIO CUES. Model pilots rely heavily on audio cues to determine the state of health of their machine. HELIX presents engine and rotor sounds, each changing with engine & rotor speed and power settings, attenuated with distance, and frequency shifted for accurate Doppler effect. Experience has shown that in the limited visual-cue environment of a simulation, good audio cues can compensate for lack of visual information. For example, during autorotation the rotor speed is primarily monitored visually by the pilot. Replacing this cue by an audible indication of rotor speed in HELIX proves not to be a distraction, and pilots often comment on the lack of rotor noise when returning to the real model.


(a) Beta-XS Helicopter

(b) Colt Aircraft

Figure 2: Typical Polygon Models

FLIGHT MODEL

HELIX presents a major problem in the design of flight model equations. The operating envelope is very wide (high angular rates, high sideslip and incidence angles are encountered at all flight speeds), and very little data is available on such conditions in models. In addition, the limited ability of the host computer dictates that the computationally simplest equations must be used wherever possible. With this contradiciton in mind, the acid test for the development of the HELIX equations has been whether they improve the response of the model *as perceived by the pilot*. Hence emphasis on accurate trim positions, long period and very high frequency dynamics is reduced in favour of reproducing accurate responses within the bandwidth of the pilot. This typically spans the range 0.1 Hz to about 5 Hz, the lower end being largely speed reponse to power and attitude changes, and the upper end being angular rate responses to control inputs.

The model equations are executed at a fixed frame rate of 36.4 Hz. This frequency is twice the system timer interrupt rate on a PC and is simplifies the software interface to the operating system. Rectangular (Euler) numerical integration is performed on the twelve rigid-body dynamics states, plus three additional states for the transmission.

The HELIX helicopter flight model is fully parameterised with about 60 independent parameters, and different models can be simulated simply by loading a new parameter file.

MODEL STRUCTURE

The model equations compute all the forces and moments applied to the airframe from thrust, aerodynamics and undercarriage sources, and then pass these totals to a module which implements the standard six degree-of-freedom rigid body equations of motion (see Stevens & Lewis [1]). Translational speeds are integrated in aircraft body axes, and then transformed to earth-fixed axes for integration to determine displacements. The angular equations are implemented use a four-parameter quaternion method which avoids the singularity associated with 90° pitch attitude, where yaw and roll angles become indistinguishable.

To simplify the angular equations somewhat, the assumption is made that the model has at least one plane of symmetry, i.e. the cross-products of inertia I_{yz} and I_{xy} are zero.

MAIN ROTOR. The aerodynamics of a helicopter rotor are exceedingly complex, since it is required not just to support the weight of the helicopter, but also to provide the major control imputs for manoeuvreing. This is achieved by varying the lift distribution around the rotor disk through cyclic and collective blade pitch changes. The subject is well covered by Prouty [2] and Newman [3], and the HELIX model follows the classical momentum methods of these, which is the only technique feasible in real-time on a PC.

This comprises calculation of thrust and drag torque coefficients based on the aerodynamic conditions at the rotor. The thrust vector F is considered to be deflected through an angle β (see figure 3 considering the rolling plane only, as an example). The vertical offset of the rotor head from the c.g. h generates the pitching and rolling moments from the main rotor, together with other terms to account for offset flapping hinges and damping effects. An additional source of damping is the flybar, which essentially operates as a two axis rate feedback stability augmentation system.

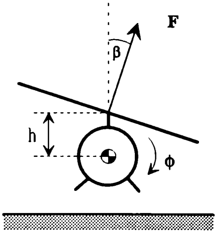


Figure 3: Helicopter Thrust Vector Model

Ground effect and translational lift (extra rotor lift as flight speed is increased) are incorporated, since these introduce quite noticable trim changes.

The rotor speed is dynamically modelled, with the driving torque being the difference between the transmission and the rotor drag torques. Accurate response of rotor speed has been found essential to the model simulation, as models are not flown at constant rotor speed, unlike the full-size counterparts. Accurate calculation of the drag torque is important, since the simulation must be capable of correct operation during autorotation. This is a most important aspect of HELIX, since engine failures are not uncommon with real models, and a simulation must be able to provide effective practice for a model pilot. The drag torque is computed via a single blade-element approximation, and the drag forces on the elements are transformed from wind to shaft axes. This allows the drag torque to go positive, indicating that the rotor is operating in autorotation mode.

TAIL ROTOR. The tail rotor is deesigned to provide a counter torque to the transmission driving the main rotor. This is achieved by applying a lateral force some distance from the c.g., and the magnitude of this is varied by changing the pitch of the tail rotor blades. The force produced varies considerably with the inflow, and hence the tail rotor provides significant yaw damping, and weathercock stability.

FUSELAGE. This is not designed to generate any useful aerodynamic forces, and is hence modelled as a bluff body with independant drag coefficients in forward and lateral directions. The vertical drag is not modelled since the dominant copmonent in this is the main rotor.

EMPENNAGE. Most model helicopters possess a horizontal taiplane and a vertical fin. These surfaces are quite small and are modelled as providing lifting (and lateral) forces only. Since these are somewhat displaced from the c.g., the resulting forces apply pitching and yawing moments, and provide restoring stability and damping in each of these axes.

ENGINE & TRANSMISSION. The power transmission path in a model helicopter is quite different from the full-size version. The engine output (usually a 2-cycle methanol engine of about 10cc) is first taken through a centrifugal clutch, which allows the engine to be started without engaging the rotors. The transmission then goes on through severl gearing stages to drive the tail and main rotors. Just before the final drive to the main rotor however, a second clutch is installed which is a one-way device, preventing the rotor from back-driving the engine. This device allows the rotor to autorotate after engine failure without being slowed by back-driving the engine, which also means that the tail rotor is not driven during autorotation. This is an important feature, since the tail rotor consumes up to 25% of the total power requirements, which is a much larger fraction than on full-size machines. Directional control during autorotation is acheived through maintaining some forward speed and manoeuvreing through cyclic pitch changes. The tail rotor slows down very quickly, and in competition flying, marks are deducted if the tail rotor is not completely stopped during an autorotational landing!

The clutches present a modelling problem for simulation, since they represent a discrete step in characteristics at some predefined condition, that is jumps in the value of moment of inertia when a clutch engages to represent the fact that two previously separate components are suddenly linked. HELIX tackles this by considering the drive train to be three independant shafts (engine, tail and main rotors) linked by friction clutches which provide drive torque only be means of slip. The clutch friction coefficients can then be tuned on-line (by making changes to the parameter files) in order to provide the best response.

The engine is modelled as generating a torque proportional to fuel flow, with a friction torque proportional to the square of the speed. This allows a good match to published engine data to be achieved, with the characteristic parabola of the speed/torque curve.

UNDERCARRIAGE. One of the most difficult aspects of model helicopter flying is landing. The model must be placed gently on the ground with vertical and horizontal velocities as low as possible, in a level attitude and with virtually zero angular rates. Exceeding any of these parameters will usually result in destruction of the helicopter, but in many cases the model will be badly perturbed by the undercarriage ground reaction but the situation might be recoverable. HELIX models the response of the undercarriage using simple spring-damper models for both translational and angular motions. The characteristics are quite different in each of the responses, and the yaw motion additionally includes a static friction term.

The undercarriage simulation poses a problem which arises due to the low mass of the helicopter and high strut stiffness. This creates dynamic modes which are higher in frequency than can be supported with a relatively low frame rate. The solution adopted in HELIX is to artificially soften the spring and damping forces to reduce the mode frequencies. It has been found that provided these parameters are carefully selected they do not attract adverse comments from pilots.

VERIFICATION & VALIDATION

Very little data is published on small rotors, especially for complete models where aerodynamic interactions between rotors and fuselage components is considerable. Hence a "response-centred" approach to validation has been adopted to date. This principally centres around presenting a large selection of models spanning small changes in a few key parameters to a sample of experienced model pilots and collecting their responses. The pilots are not told which parameters are under study, but are instructed to perform certain simple manoeuvres and give a score on some particular aspect of the response. This is in essence similar to the "mission-task-element" decomposition of handling qualities requirements advocated in current requirements documents for full-size vehicles (see Padfield [4] and ADS-33C [5]).

Currently, a sensor fit and datalogging equipment are being developed for an *Omega Pro* model helicopter, and it is hoped that system identification techniques will allow a more quantitative approach to be developed. The flight recorder in HELIX can also export data in a form suitable for mathematical analysis in MATLAB or MATHEMATICA (figures 4 and 5), and these tools are to be used as the basis for verification and validation.

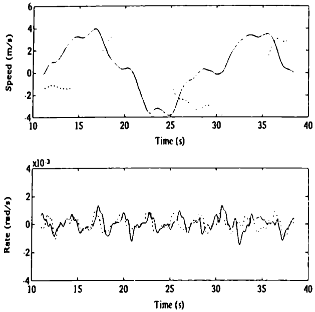


Figure 4: MATLAB Angular & Translational Rates

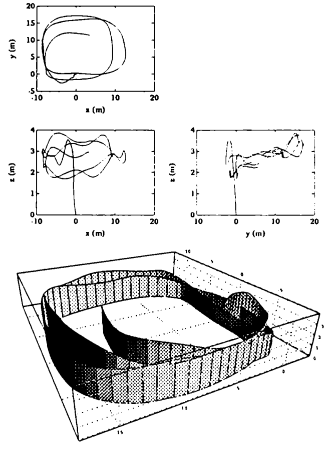


Figure 5: Trajectory Plots

CONCLUSIONS

HELIX has enjoyed considerable commercial success (used in over thirty countries) and the customer responses indicate that it is being used mostly as a serious training aid, and not simply relegated to the status of another video game. This is interpreted as justification for the simplifications and design decisions used in the development of the system.

The increases in computational ability of PCs will open the door to further development of the graphics display and flight model equations. The latter will in the near future be supported by flight trials data from real models.

REFERENCES

[1] B L Stevens and F L Lewis. *Aircraft Control and Simulation*. Wiley, 1992.

[2] R W Prouty. *Helicopter Performance, Stability and Control.* Prindle, Weber and Schmidt, 1986.

[3] S Newman. *The Foundations of Helicopter Flight.* Arnold, 1994.

[4] G D Padfield. *Helicopter Flight Dynamics.* Blackwell Science, 1996.

[5] Handling Qualities Requirements for Military Rotorcraft. Aeronautical Design Standard ADS-33C, United States Army Aviation Systems Command, St Louis, MO, August 1989.

AIAA-96-3484-CP

A RULE BASED APPROACH TO FAST TIME SIMULATION OF AN AIR TRAFFIC CONTROLLER

D. N. Siksik Eng., R. Prager, J. Muise
CAE Electronics Ltd.
St. Laurent, Quebec, Canada

Abstract

Tactical and non-tactical flight simulation has seen a rise, over the past few years, in the use of synthetic environments. With their introduction to flight simulation, new requirements to support the functionality of synthetic environments are being brought to light. One such requirement is the ability to execute training and research scenarios in fast-time, as opposed to real-time. This requirement is founded in a desire to execute simulations as quickly as possible in order to provide scenario verification for training applications or to provide various simulation results of "what-if" scenarios for research applications.

Introduction

As airspace becomes increasingly congested and as the workload of air traffic controllers rises, the role of appropriate and effective procedures cannot be underemphasized.

The Canadian Airspace Management SIMulator (CAMSIM) project currently being built by CAE Electronics Ltd. for Transport Canada addresses concerns of the role of procedures now in place and provides the means for tailoring air traffic control solutions into the next century.

The CAMSIM programme consists of both a real time and a fast time simulator.

CAMSIM Real Time

The real time simulator is designed to provide a facility to both test out new procedures and to evaluate new technology as well as to train and certify air traffic control (ATC) personnel.

This simulator consists of manned ATC and pilot workstations which allows control of simulated aircraft during evaluation and training. Exercises are created on a separate workstation using an offline databasing system.

The real time simulator design stresses the role of ATC communications by reproducing typical ATC communications facilities. The representative behaviour of the simulated aircraft is also critical and, to this end, aircraft are modelled to within 5% of their performance envelope.

CAMSIM Fast Time

As opposed to its real time counterpart, the fast time simulator is geared towards the evaluation of current standards and procedures as well as the development of new ones. This simulator will be used in determining procedures based upon a global picture of the air traffic situation over a long period of time. A typical exercise may involve monitoring air traffic through several sectors across the country during a time span of 24 hrs.

The fast time simulator does not include humans in the loop as does the real time. The architecture therefore consists of a single workstation upon which the exercise is run. As with the real time simulator, fast time includes an exercise preparation facility and retains the high level of aircraft modelling fidelity.

This paper presents the design approach taken to create the fast time simulator.

First discussed is the concept of fast time and the method used to achieve a fast time simulation. The fast time components are then presented in order to provide a global view of the design.

The discussion then focuses on a critical area of the fast time simulator: the Air Traffic Controller Model. Since fast time precludes humans in the loop, it was necessary to develop a representative model for the air traffic controller.

The controller design, which is based upon an artificial intelligence rule based approach to simulating decisions, is presented.

In closing, future trends and expected developments are discussed.

The Jump to Fast Time

Functionality

The CAMSIM ATC simulator achieves its goals by providing simulation in real time. As discussed above, it is possible to conduct research on potential ATC procedures in real time.

Many procedures, however, are effective over a long period of time (greater than 24 hours) and the effect of several organizational structures, such as sectorizations, is best considered over the span of several hours or days, during which both high and low air traffic patterns develop.

In order to accommodate the requirement to simulate an air traffic environment over several hours and in order to avoid the necessity of excessively long exercise durations and continuous human intervention, the concept of fast time simulation is introduced.

Fast time simulation allows a many to one ratio of simulated time versus actual time (as opposed to real time simulation, which provides a one to one ratio). This ability implies that a 24 hour simulated exercise run in fast time will actually take only a few minutes.

Using fast time, it becomes possible to efficiently execute scenarios which encompass hours or days in order to analyse the effect of procedures and organization throughout periods of fluctuating traffic.

Achieving Fast Time

Fast time simulation may be achieved by increasing the time constant of the simulation models, by processing the models in a computer intensive manner as well as by implementing event based processes. The theory behind each of these is discussed below.

Increasing the Time Constant:

The simulation time constant represents the amount of simulated time which a model considers itself to take.

A continuous process will be called and will re-execute without interruption at every clock tick at an iteration rate of i Hz. This implies that the continuous process has $1/i$ seconds in which to complete all calculations.

A real time simulation being represented by a continuous process considers the simulation time passed during one iteration as being equal to the iteration period $P = 1/i$ seconds. A fast time of n times (nx) can be achieved by considering the simulation time as being 2x, 4x, ..., nx,.... of the iteration period or nP.

The result is that models represented by the simulation consider nP seconds of simulation time to have passed for every period of execution. The larger n is, the greater the time acceleration.

Computer Intensive Processing:

A continuous process may execute repeatedly in a CPU intensive manner. This may be contrasted with the real time continuous process discussed above. The latter has a minimum of P seconds to execute and typically will take less time so that the computer has the appropriate spare time for user input and intervention during the real time exercise.

Computer intensive processing implies that the process is restarted as soon as it has finished. There is no allowance for spare time and CPU processing is maximized. This applies equally to continuous and event based processes.

To illustrate, consider an example in which it is required to simulate 1000 aircraft over a period of 24 hours. It may be assumed that this is equivalent to an average of 150 aircraft in the air at all time.

If one aircraft requires 1 msec of computer time, then 150 aircraft require 150 msec. If the time constant is four, then 150msec of computer time is equivalent to 4 seconds of simulated time.

The speed up factor is 4/150 msec or 26.7. For a 24 hour exercise, this translates to a duration of 54 minutes.

Fast Time Software Modules/Architecture

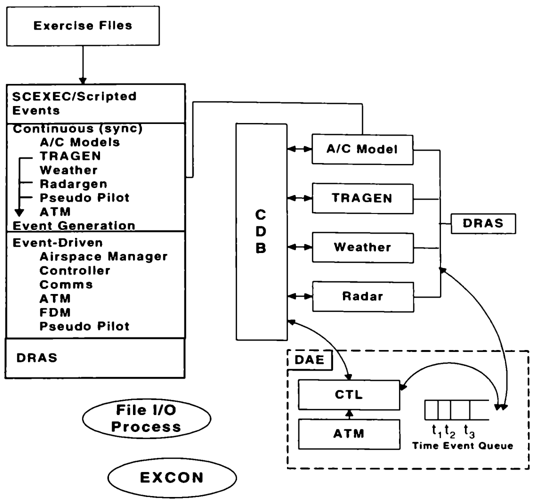


Figure 1 Fast Time Architecture

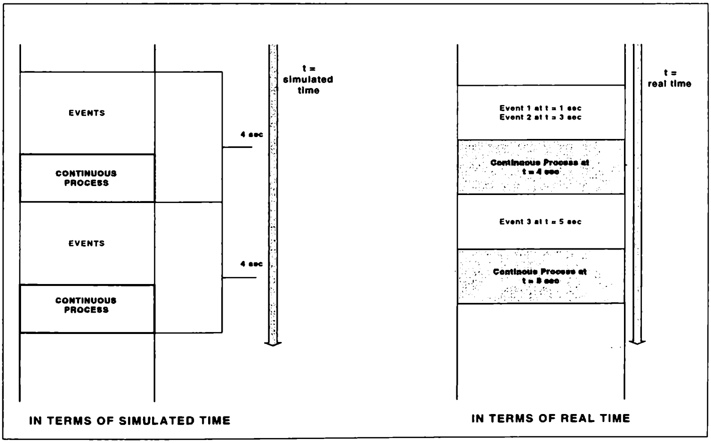


Figure 2 Simulated Time Versus Real Time Scheduling

Event Based Simulation:

Event based simulation is a third method for implementing fast time simulation. In event based systems, time is advanced from one event to the next. In this situation, the simulation clock is based upon the times at which the events occurred, as opposed to a set iteration rate or a multiple thereof.

In this approach events can be processed as they occur, removing the requirement for having to poll for their occurrence, as is the case with continuous processes. This is advantageous so long as events are not frequent. As the frequency of events increases, the overhead of event handlers can exceed the overhead of polling.

The result of this trade-off is that for processes with infrequent events, an event based approach executes in significantly less time and is therefore capable of representing a fast time simulation.

The Fast Time Simulator:

The CAMSIM fast time simulator applies all of the above strategies in order to achieve accelerated time.

Critical elements of an air traffic control simulator include the navigation and dynamics of aircraft and the control process of the air traffic controllers. The former element is well represented by a continuous process while the latter is inherently event driven.

This application includes both continuous and event based processes. The continuous element is called every four seconds of simulated time, thus providing a speed increase ratio of four.

Events representing air traffic controller behaviour are scheduled and queued, thus eliminating any polling that would be required for those events.

The fast time implementation uses a single computer process, which includes continuous and event based elements, to ensure repeatability (see figure 1).

All processing is actually executed as soon as possible in a CPU intensive manner. Computer resources are therefore fully exploited (see figure 2).

Fast Time Simulator Components

The fast time simulator may be broken down into a number of components. Those based upon continuous processes are also part of the real time simulator while those which are event based are newly designed for the fast time implementation. The major components are described below.

Scenario Executive:

The Scenario Executive is the principle element of control during the initialization of a fast time exercise.

The Scenario Executive will load flight plan routes as well as call the appropriate conversion program in order to provide the actual sector boundary crossing times for those routes.

The Scenario Executive calls a ground delay routine in order to schedule flights. For example, an aircraft bound for Chicago will not be scheduled for take-off if it is known that, at arrival, Chicago will not be able to accommodate it.

Control during the simulation itself is provided by scripted events. These events are user created and are time based. They provide a means to automate runway closings, sector consolidations, radar failures, etc.

Aircraft Model:

In order to simulate aircraft flight along a flight plan, the Aircraft Model[1] component provides modelling of the aircraft dynamics.

Using a customized building tool, it is possible to generate high fidelity models from a limited set of parameters. These parameters may be obtained from such sources as Jane's All the World. Any aircraft type commonly flying within Canadian airspace may be simulated.

Trajectory Generation (TRAGEN):

Navigation along a pre-set flight plan is provided by the Trajectory Generation component. Since flight plans are specified via navigational aids, TRAGEN uses a converted flight plan for which actual times are provided at positions such as beacons and sector boundaries.

Commands to TRAGEN from the air traffic controller are provided via an interface called the Pseudo-Pilot. The Pseudo-Pilot implements controller commands such as climb and descend, vector or position report requests.

Radar and Weather:

The Radar component provides the simulation with a defined radar coverage area and with aircraft positions within that area.

Aircraft tracks with corresponding flight database entries and unique search secondary radar codes are automatically correlated to their radar image. The air traffic controller model manually correlates all other aircraft.

Air Traffic Manager:

Air Traffic Manager functions exist in order to allow automatic processing of basic air traffic control functions, thereby reducing the workload of the controller. The Air Traffic Manager includes the following functions:

Monitor and Alert Program:

Airspace availability is defined, for resources or critical points such as runways, airports, sectors and fixes, by values of capacity and demand.

The Monitor and Alert program considers current traffic levels against predicted future levels for these resources. An alert is generated if saturation will occur.

Terminal Metering Fix Program:

When arriving at an airport, aircraft must be sequenced in order to allow the minimum separations at the runway threshold. They may also be sequenced on a first-come/first-served basis, based upon aircraft type or the target runway.

The Terminal Metering Fix program calculates the desired arrival sequence as well as the corresponding arrival time. This information is transferred to the controller to advise him of the incoming aircraft.

Ground Delay Program:

The Ground Delay program implements delays in aircraft routes for aircraft which are still at their origin and for which aerial holding is predicted because of overloaded or unavailable resources at the destination.

The Air Traffic Manager functions may be activated for user selected resources, such as runways or airports based upon exercise time or specific events.

While ground delay is implemented as part of the flight plan injection into the scenario via the Scenario Executive, the Monitor Alert and Metering Fix programs result in advisories which are routed to the controller.

Communications:

The communications process undertaken by the controller with respect to aircraft as well as other controllers is modelled. The model considers a number of factors such as radio noise and interference in order to calculate a probability of successful communications based upon a normal distribution curve.

Air traffic control stations include several means of communications, such as radio, interphone and hotlines. Each of these devices is considered in the communications task, taking into account the role of each device and the extent of user loading.

Exercise Preparation and Data Recording:

An effective and easy to use exercise preparation utility is essential in the creation of complex large scale scenarios. Exercise Preparation is a data entry system, based upon the SYBASE database and X-Windows, which allows the user to create fast time scenarios in an object based manner. This approach allows for the parametrization of many simulation values as well as for the re-use of such elements as flight plans.

For experiments involving large number of aircraft, a stochastic flight plan generation utility is included with the intention of reducing the time required to generate an exercise.

The exercise execution in fast time is not monitored by the user. Rather, the exercise is recorded and may be played back at any speed. A map-style two dimensional display is used to visualize the playback process. Data recorded during playback may be exported to any databasing system for analysis.

The Air Traffic Controller Model

The fast time simulator, having no humans in the loop, must simulate the function of air traffic controllers. Since a potential exercise may include many controllers and since their tasks are event based by nature, an object-oriented event based approach was adopted. Controller decisions, on the other hand, are based upon published procedures which are very rule oriented. A rule-based approach towards representing controller decisions was therefore appropriate.

The following sections give an overview of the process of event driven simulation as applied to the controller model, the controller simulation with respect to the tasks and events defined and finally the representation of the controller's application of aircraft separation standards via rules.

Event Driven Simulation

An event-driven simulation consists of a number of agents and a dispatcher. Agents correspond to real-world individuals or entities such as pilots, controllers, or traffic manager. Each individual is typically modelled as a single agent. Other agents in a simulation may correspond to less distinct roles, such as an "airspace manager" which is responsible for closing runways based on weather.

An agent will interact with other agents and with the dispatcher. These interactions are kept distinct, as illustrated in Figure 3. The dispatcher is not involved in passing events and agents do not directly activate each other. Each agent is highly autonomous, when an agent gets a new event it will decide how and when to respond and inform the dispatcher of when it needs to be activated.

Another component of the discrete-event simulation design is the directory. The directory maintains relationships amongst agents and other objects. It is accessed by the event dispatch mechanism so that all events are sent to the appropriate destination agent.

Discrete Time Dispatcher

The dispatcher maintains the discrete-time clock and is responsible for sharing CPU time among agents to achieve the effect of concurrent execution of agents. The main agenda contains at most one entry per agent, the next time at which the agent must be run, as determined by the individual agent. Each agenda entry consists of a pair of variables: <time, agent>. An agenda entry <t1, A1> implies agent A1 will be run at time t1.

The queue is sorted by increasing order time values. After an agent A2 runs it returns a new time value, say t2, the time when it must run next. The new entry <t2, A2> is inserted back in the agenda, in the appropriate place, by the DTD. The dispatcher consists primarily of a loop which pops entries off the queue, advances the clock, and runs individual agents. Each time the dispatcher runs an agent new events can be created, resulting in new <time, agent> entries being added to the queue. There can be multiple entries with the same time value, if so these are run in a FIFO order.

Agents

Agents are event-driven, concurrent, and asynchronous. Agents communicate amongst themselves by sending events and only by sending events. Each agent maintains a local event queue to manage its own workload. Different types of agents are defined by the set of events they can receive and their responses to those events. Simpler forms of agents can be defined but are not discussed further in this paper.

An agent's response to an event typically consists of several distinct "tasks". An agent's tasks correspond directly to the different tasks/actions/abilities of a real controller.

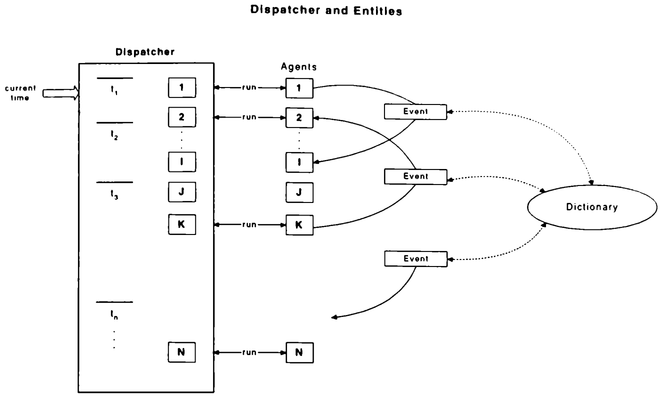


Figure 3 Overview of Dispatcher and Entities

Rule #1

```
- (defrule vertical-separation-when passing
fs:(FIND-SEPARATION flight1 = flt1 flight2 = flt2)
(RECIPROCAL-TRACK flight 1 = flt1 flight2 = flt2)

→

{ passing-time = estimated-passing-time (flt1, flt2);
before-passing = passing time - 10 minutes
after-passing = passing-time + 10 minutes
if intervals-overlap (fs, before-passing, after-passing)
    {(end-separation longitudinal, flt1, flt2, before-passing)
     (start-separation longitudinal, flt1, flt2, 10 minutes),
     after-passing
     (need-separation vertical, flt1, flt2, before-passing,
       after-passing)
})
```

Rule #2

```
- (defrule cross-tracks-separation-484.11AB default
fs:(FIND-SEPARATION flight1 = flt1 flight2 = flt2)
(CROSSING-AT-POINT point = point
   track1  = track1 track2 = track2)
{flt 1->includes(track1) && flt2 ->includes(track2)}

→

 {if (flt1->navaid-spacking (track1) <= 40min &&
     flt2->navaid-spacing(track2) <= 40min
   then duration = 10 minutes
   else duration = 15 minutes
   (assert (RESERVATION center = point
          radius = duration units = minutes
          flight1 = flt1 flight2 = flt2)))
```

Figure 4 Sample Separation Rules

Tasks model the flow of events in a controller's work. An input event, for example an altitude change request from a pilot, sets off a cascade of tasks, for example updating flight plans, checking for separation, coordinating with other controllers, issuing clearances, re-planing flights, etc. Tasks pass events between each other and there may even be loops. For example if an altitude change is not acceptable to the following controller, then further planning and separation checks must occur to find an altitude change acceptable to the pilot and both controllers.

Controller Simulation

Each controller is modelled as an agent in the discrete-event simulation architecture.

There will be several types of controller (e.g. oceanic, en-route, terminal, etc.) Each type of controller can have unique logic for planning, separations, or other controller functions. There will typically be many instances of each type of controller in a fast time simulation exercise. Every controller will read events from its event queue and process them, by scheduling certain tasks. Simple fixed values will be used to account for controller work-load. Processing of some events will involve running rules.

A controller's most basic function is to ensure all flights in his/her sector are safely separated from each other at all times. A controller always plans ahead, predicting potential losses of separation (or *conflicts*) early enough to plan and implement a resolution of the conflict. As events are received the controller updates his/her "picture" of each flight's progression through the sector and checks separation rules on the situation as it is expected to be in the future.

Controller Events

Every controller must be able to respond to many types of events, from many sources. For example, the Pseudo-Pilot will send events corresponding to radio contact, position reports (multiple sub-types), clearance requests and responses to controller queries. Events sent between controllers include various co-ordinations, estimates, point-outs and setting flight restrictions.

Flight Profiles and Conflicts

Each controller is concerned with a single control sector, or simply *sector*, the flights in the sector, and some in-bound flights from adjacent sectors. The *state* of the sector, known by controllers as the "picture", consists of three kinds of information about flights: profiles, conflicts, and flight states.

Every flight in the sector follows some route, or *profile*, which is agreed to by the controller and the pilot. The controller will generally try to allow a flight to follow its filed flight plan as closely as possible, but various over-riding rules or restrictions may have to be applied.

There are three basic versions of a flight profile:

1. The *cleared profile* represents the profile which has been agreed to by the pilot and controller.

2. The *planned profile* is modified to resolve a potential loss of separation, to apply unit or sector procedures, to meet restrictions which are imposed by adjacent sectors or to comply with advisories from Air Traffic Manager (ATM).

3. The *working profile* represents the controller's trial solutions to potential problems.

The controller re-examines profiles when new events (eg: position reports) arrive. Several situations require a controller to take action, including a potential loss of separation (discussed below), a mismatch between planned and cleared profiles (implies new clearances for the pilot) or a conflict between a profile and some restrictions specific to a sector (eg: due to mountainous terrain).

A controller will look ahead to the future sector state to detect conflicts and each conflict is tagged with the time when it may occur. Normally a controller will take some actions to ensure future, or potential, conflicts never actually occur. The exception is ATM advisories, which are not compulsory.

Controller Tasks

The controller spends his/her time doing the following tasks. Each task has an associated workload which is accumulated for data recording. Only one task at a time will be active. An incoming event will trigger one or more tasks.

Communications In-coming: Handles in-coming radio and phone (hot-line or inter-phone) messages and produces a stream of events to Monitoring task.

Monitoring: Examines all in-coming events and determines what the controller's response.

Separation Checks: Runs separation rules to decide what form of separation will be maintained between each pair of aircraft within the sector.

Planning: Maintains the planned profile for each flight.

Conflict Resolution: Formulate a resolution (ie: a new planned profile) to avoid any loss of separation.

Procedure Handling: Handles most "routine" procedural operations.

Communications Out-going: Handles out-going radio and phone (hot-line or inter-phone) messages and produces a stream of events to other agents.

Separation Rules

A distinctive aspect of the controller is the rule-based design for ensuring positive separation between pairs of aircraft. Separation standards are based on ICAO documents with variations as defined by national authorities (e.g. in Canada the Air Traffic Services branch of Transport Canada). Each rule describes a certain situation and the applicable separation standards for that situation. A separation standard between two aircraft may be expressed in terms of a difference in altitudes (e.g. separate by 2000 ft.), time (e.g. 15 minutes flying time apart on the same track) or horizontal distance (e.g. 1 mile from a holding area).

These rules are implemented, in the SEPARATE task, using a forward-chaining rule-based package known as ILOG Rules from ILOG Inc. Forward chaining rules are used so that as new events come in to the controller only the minimal number of rules which are possibly affected by the new event will be re-checked. (This relies on the RETE algorithm used in ILOG Rules.)

ILOG Rules was chosen since it offers a logically complete rule language, support for multiple agents, tight integration with C++, and code-generation and the RETE algorithm for high performance.

Separation Plan

A separation is a specification of a distance which must be maintained between a pair of aircraft in order for them to be considered not in conflict. A separation is always associated to a specific pair of aircraft. A separation is a distance in a single dimension (vertical, longitudinal, lateral), or geographic (pertains to aircraft on different airways) or horizontal (pertains to aircraft on tracks other than defined airways).

A separation also has an associated begin time and end times. These give the time interval for which the controller expects the separation will be safe. Since a controller plans ahead the begin time will typically be some time in the future with respect to the time (simulation time) when the controller creates the separation.

The separation plan is the set of separations which have been planned by the controller, which will guarantee separation, and the assumptions upon which the plan depends, i.e. the conditions which existed at the time the plan was made.

Sample Rules

In general, controller separation rules adhere to the following general structure:

IF - *the left hand side*
flights match some pattern involving routes, navaids, etc.

THEN - *the right hand side*
calculate start/end times of separation interval
calculate amount of separation (time, distance, etc.)
create/modify separation intervals

Two examples follow. In the examples, an expression "(FIND-SEPARATION...)" indicates an object which directs the rules to find a positive separation between two flights.

In the first example rule (see figure 4), two aircraft, flt1 and flt2 are on a reciprocal track and are estimated to pass each other at a time, *passing-time*. The *intervals - overlap* test ensures that these separations overlap, in time, with the interval of interest given by the FIND-SEPARATION object. The logic of this rule is that from 10 minutes before, to 10 minutes after they cross, flt1 and flt2 must have vertical separation. Before and after that, they will have longitudinal separation.

The second example (see figure 4) concerns two flights, flt1 and flt2, whose tracks cross at a "hot spot". Depending on the spacing of navaids, either 10 or 15 minutes horizontal separation is required. This is handled by creating a RESERVATION object, centered on the crossing point, which directs the controller to plan on only one flight at a time in an area within the given radius.

Conclusions

This paper has reviewed the implementation of fast time processing for an air traffic control simulation as well as the design of an AI based controller.

It was noted that fast time may be achieved through a time constant change, intensive processing or events. Event scheduling was discussed in detail and it was shown that this is an effective method to represent the tasks of a controller. The decision process of the controller was reviewed as being provided via rules.

Applications of these technologies include providing fast time capability to other simulations as well as providing an automatic controller for military or civilian flight simulators.

References

1. Hached, M., Beauchesne, G., Aircraft Performance Modelling for Air Traffic Control Simulation Applications, AIAA Conference on Simulation and Technology, Baltimore, Maryland, August 1995.

2. Transport Canada, Aviation, Air Traffic Control Manual of Operations, April 1993.

Acknowledgements

The authors would like to acknowledge the dedication and hard work of the entire Fast Time team, without whom this paper could not have been written.

AIAA-96-3486-CP

THE PERENNIAL PROBLEM OF SIMULATOR DATA: A SELF-INFLICTED WOUND

By: Captain Erik Reed Mohn
SAS Flight Academy

Abstract:

Since the first flight simulator was produced, what and how to program it has been debated. The debate today is not so much about what to put into it, that is pretty much taken care of by the current regulations and practises. The focus today is rather on how to improve on the quality of what is already there in order to improve on the simulation itself.

So what are the problems?

1. There is as yet no industry consensus on how to build simulators, or rather, how to simulate the aircraft. Options range from as many hardware (black) boxes as possible to as pure a software solution as possible.

2. Some aircraft manufacturers and simulator operators are engageing in questionable business practises in order to restrict competition.

3. The process of obtaining whatever data needed for the solution a simulator customer chooses is grossly inefficient.

4. This exposes us to unnecessary risk and expense in obtaining data for our simulators.

5. The industry has not obtained consensus either among operators or inside individual companies on how to handle this.

This state of affairs is unsatisfactory in the extreme and these unresolved questions are the self-inflicted wound.

The paper will attempt to go through all these problems and also propose a number of actions which are available in order to alleviate them.

This industry has indeed come a very long way since the FAA´s advanced simulation plan saw the light of day in the late 70s.

Tens of thousands of pilots in the US have since then been trained through what is known as zero-flight time programs, and it has really worked like a charm. At least the industry agrees on that, and such programs are now belatedly spreading to Europe and will undoubtedly in time spread to the rest of the world.

When you can demonstrate that a pilot can be trained to operate an advanced aircraft safely through the use of simulators only, you have obviously demonstrated that whatever is inside those devices have a reasonably high quality. Otherwise the control systems in place for catching substandard performance among pilots would have kicked in long time ago and this practise stopped dead in its tracks.

The question is whether current practises will be enough for the future.

I believe they will not.

Nobody, as far as I know, is seriously concerned with the quality of the performance or handling characteristics data in our simulators. They can always be improved, but as far as normal operations are concerned, we are pretty happy.

We are so satisfied with this side of the data in fact, that at the initiative of Boeing, the issue of using data from engineering simulators as an alternative to flight test data to program and validate minor changes in an aircraft is now in a prototype activity phase. The prototype program is the new generation 737 for changes in body length. Airbus is working on a proposal to try the same for validation of changes to control laws in computer controlled aircraft.

If these activities are successful, and I personally have no doubt that they will be, this should lead gradually to an expansion of what constitutes acceptable changes which may be validated by data derived from other sources than flight tests.

The Royal Aeronautical Society's annual spring simulator conference unanimously endorsed this trial program on May 17th after an international working group had been addressing this and other data problems through a series of meetings over the last two years.

This should, when successful, lead to faster updates and better simulation earlier in the life of a new aircraft program.

Such a program will not, however, lead to immediate and dramatic decreases in the cost of data, but it should over time decrease the aircraft manufacturer's costs and hopefully filter down to us poor guys who buy from them.

And this connects us neatly into the contentious subjects of price and availability of data.

It is always available. Somehow. As far as I know, there are no large aircraft programs today which does not in one way or another come up with a level D data package. Sometimes in a manufacturer specified format we would rather not see. Sometimes the prices charged for data come uncomfortably close to extortion But it is available. Most of the time.

Not always to all, though. There have been documented cases where so many hurdles have been put in potential simulator operators way that they have all but given up in the end. These hurdles have been of a monetary, contractual and plain obstructionist nature.

Most of the people in the industry find this way of doing business unacceptable, and the RAeS working group also presented to the Society on May 17th a resolution squarely aimed at such reprehensible practises.

It too was unanimously adopted, but whether this will have any effect is anybody's guess, but the practitioners have at least been put on notice.

I will not say too much about how people should go about specifying how their aircraft should be simulated. Suffice it to say that I firmly believe that the days of reverse engineering for simulator data is over.

Todays integrated cockpits are very complicated beasts. Unless we either communicate perfectly with hardware or we rehost, port or whatever you want to call it, we face insurmountable problems in trusting that the simulation is correct. Correct in this context meaning that you get the same reaction in the simulator as you would in the aircraft, from the same action.

Companies making the integrated avionics are now faced with a problem. It is apparent that most people today will opt for the rehosting or porting solution. This will deprive the avionics manufacturers of sales of their not-so-cheap black boxes for simulators. That they wish to keep up their income from their business is only natural, but I sometimes get the feeling that the data availability and pricing policies at work here are not benign.

Because the aircraft manufacturer has selected an avionics supplier as a single source supplier on his aircraft, we, the secondary buyers of the supplier's equipment or data are put at a distinct disadvantage in negotiations.

The deal is done. We have bought the aircraft and now we need a simulator. We can go nowhere else but to the vendor for the equipment or data we need. Squeeze time.

We must also to take a look at how the process of obtaining data for simulators work today.

It is not an easy process to unravel, even for insiders, because there are conflicting interests at work, and protection of income instincts are as strong here as in any other part of life.

Normally it works something like this: we identify a need for a simulator and contact the simulator manufacturers for bids. A winning bid is selected and the data can either be specified as BFE or SFE. These details vary with different companies and their relation first of all with the aircraft manufacturer.

The aircraft manufacturer is obviously important here, and they all have a data package to sell. That data package does not normally contain all the data or parts needed to build a simulator, since data and parts for vendor equipment is in most cases missing.

Now, you cannot build a simulator without simulating equipment supplied to the aircraft manufacturer by vendors. That includes such heavy items as autoflight systems, engines, flight management systems and a host of others absolutely necessary for the aircraft and simulator.

And, on top of this, the aircraft manufacturers, if they are to be believed, do not carry a lot of clout with the vendors.

The nornal routine is then to let the simulator manufacturer negotiate with the vendors after the methods of simulation are specified, and then pass the bill to us, the buyers or operators.

Not transparent and not good.

Kip Caudrey of CAE gave a paper at the RAeS meeting in May, and it is useful to look at some of the figures he came up with.

I am sure some of you know Dieter Hass of Lufthansa. He was quoted by Kip as saying: "In July 1971 the least expensive Lufthansa flight from Frankfurt to New York was DM 1362. In 1994 it was DM 1299. The dry flight simulator cost per hour was DM 470 in 1971 and DM 1400 in 1994, based on 4600 hours per year."

Add to this the fact that roughly 40% of the cost of a full flight simulator are related to the purchase of aircraft hardware, avionics boxes and data packages.

We may all come up with different numbers, but I belive the basic fact will remain the same; we are faced with spiraling costs in our part of the industry.

But the sky is never gray above the clouds, if you can only get there. It may not be easy, but I believe it can be done.

Given the fact that there are but a handful of aircraft manufacturers and equipment vendors, and that most of them seem fairly satisfied with status quo and see no compelling reason to rock the boat, it follows that it is the simulator and aircraft buyers and operators who must unite in order to force changes.

Not very strange that, since we are the ones who are not happy, but there are an enormous number of us and not easy to get the attention of everybody and then coordinated action.

Nevertheless, international cooperation in the field of simulation has taken giant steps in the near past, however, and is progressing every day.

The RAeS took the initiative to have technical simulator qualification standards harmonized throughout the world.

This has met with unqualified success now that the ICAO handbook is adopted, the JAR STD Part 1A is published and the 120-40C is or will be published shortly. All these documents have the same technical requirements for the the two highest levels of simulators.

These efforts pave the way for the holy grail of this part of the industry, the mutual recognition of simulator qualifications all over the world. It will not be an reality today or tomorrow, but the foundation is laid.

Another successful cooperative effort has been the RAeS working group already referred to. In addition to what I have already said about the results from this group's work, it reopened an IATA working group to revise the IATA Flight Simulator Design & Performance Data Requirements document.

This was done successfully in Seattle in April.

Although no simple solution to all our woes has presented itself, there seems to be a growing feeling not only that something should be done, but also that something actually can be done.

As far as I'm concerned, we, the buyers of aircraft, hold the only big stick the aircraft manufacturers will pay attention to: our money. Our billions, in fact. Sounds good, doesn't it?

This "weapon", if you will, is only effective when the contract for an aircraft purchase is being negotiated. It's too late when we have signed on the dotted line, because this kind of leverage is fleeting like the wind.

If you are like me, you also tend to believe a little more in the aircraft manufacturer's power over their vendors than they appear to do themselves .

Provided that these observations are correct, we have the outline of a possible solution to our problems: dump them on the aircraft manufacturer. To put a nice name on it we call it one-stop-shopping.

It is very crudely put, but it's exactly what we aim to do at SAS. Unless we don't do our homework and get bushwacked by the aircraft manufacturer's negotiating team, the next time we buy an aircraft, we will include *all* data and parts for simulators in the aircraft purchase.

We are still working out exactly how to proceed, but it may not be unthinkable to include the necessary simulators in the aircraft price.

What I cannot fathom is the resistance such ideas meet.

If I were Mr Boeing, Mr Douglas or Mr Airbus, I would insiste that this be part of the product I sold. We are in the safety business, and it is absolutely unthinkable to train pilots to fly a transport aircraft of any size today without simulators.

Since aircraft manufacturers have a sizable stake in the safe operation of their product, they ought to ease the availability of anything which enhances the quality and availability of training for their aircraft. They don't today.

It will not be easy. This will require renegotiation of contracts with vendors and a total rethink of the way we do business.

The sad fact is that training and simulation are still treated as sort of an afterthought by high level executives in our business. This includes the operators' executives, in case you thought I was bashing only everybody else.

Compared to the price and effort of building aircraft, it can be argued that data and parts for simulators are trifles and not worthy of senior management's attention.

Try to imagine what we would be doing without it, and your view may change ever so slightly.

What we are really saying is that we will insiste on this being an integral part of the contract before we agree to buy your aircraft.

You are of course perfectly free to disregard demands like this, but considering the relative worth of the items in question, we ask you to consider if it's worth it.

What we also want is for the aircraft manufacturers to tell their suppliers that their equipment does not go onboard your aircraft unless they agree to be totally cooperative in supplying data and parts for simulators.

The point of this is not really to bash anybody, but hopefully to reach out and maybe start a thought process which may make training better and our life easier in the long run.

There are actually a number of initiatives which support such ideas in the works.

The mentioned revision of the IATA data document goes several steps in this direction. The RAeS meeting in May adopted a resolution saying that data should be part of the contract between the equipment supplier and the aircraft manufacturer.

The US Air Transport Association will look into the possibility of recommending as a policy to its members that simulator data packages be delivered with the aircraft.

For my own part I will continue to argue this wherever and whenever I can. I have a complete file of aircraft leasing companies who will be introduced to this at an opportune moment. I keep the Association of European Airlines members informed and new links are being forged with the Orient Airlines Association. I also know that several airlines are quietly revising their contract on file for future use.

It must be said clearly to avoid misunderstandings that the aim of this is not primarily price reductions. We are all aware of the more than tenuous link between cost and price in this great business we're in. But on the other hand we all know that costs must be covered.

The purpose is to get a system in place which is better and more conducive to an efficient flow and availability of good data, so we can become more efficient and then the savings will come.

The collective industry needs to swallow a few camels in order to do this, but it can be done.

The walking softly and carrying a big stick should change to walking firmly and brandishing the damned thing.

Thank you for listening, ladies and gentlemen.

Contact address:

Erik Reed Mohn
Manager Governmental & External Affairs
STOOT-S
SAS Flight Academy
S-195 87 Stockholm
SWEDEN

Telephone office: +46-8-797 4694
Fax office: +46-8-797 4240
Mobile phone: +46-70-593 9740
Telephone home: +47-22 50 77 15
Fax home: +47-22 50 94 19

AIAA-96-3487-CP

ADVANCED MANEUVER TRAINING IN FULL FLIGHT SIMULATORS

Ron Shoulars
American Airlines

ABSTRACT

In recent years several air carrier accidents have occurred where the pilots were confronted with an unusual situation from which they were unable to recover. Analysis of these accidents identified areas for improvement in the training of airline crews to recover from various inflight situations. In March, 1995 an industry wide conference was held to discuss approaches to resolve these deficiencies. American Airlines developed a comprehensive academic and simulator training program which is being given to all aircrew members. Realistic implementation of the training maneuvers in the simulator proved to be both a challenging and rewarding experience. This paper will address the maneuvers developed and how they are incorporated into the training program.

INTRODUCTION

American Airlines pilot training is conducted in nineteen full flight simulators and two flight training devices. In addition three simulators are used for Eagle training and five are used primarily for contract training. These devices range from a Level A, three degree of freedom (DOF), single visual channel Boeing 707 to a state-of-the-art, six degree of freedom, Level D MD-11 with a three channel wide SPX-500 visual system. After several iterations the program was titled Advanced Aircraft Maneuvering Program (AAMP). A summary of the devices with AAMP installed is presented in Table 1. All American Airlines pilots are currently receiving the training during both initial and recurrent training. Many of the concerns as to how to implement the program without damaging the equipment will be discussed.

Although efforts were made to make the training as realistic as possible it is emphasized that the important thing is not how you get in the situation, but how you get out.

Table 1 - AAMP SIMULATORS

Aircraft	FAA Level	Motion System	Visual System
MD-11	D	6 DOF	SPX-500
A-300	C	6 DOF	SP-1T
4 x MD-80	D	6 DOF	Image 3T
1 x MD-80	D	6 DOF	SPX-500
2 x B-757	C/D	6 DOF	SPX-200 / 500
2 x B-767	C/D	6 DOF	SP-2 / Image 3T
2 x F-100	C/D	6 DOF	SPX-200 / 500
2 x DC-10	C	6 DOF	SP-1
2 x B-727	C/D	6 DOF	SP-2 / SP-3T
2 x B-727	B	3 DOF	SP-1 / SP-2

* SPX Systems are Wide Projection Visual Systems. All others are CRTs

THE MANEUVERS

To define the maneuvers to be developed the current state of the simulation was analyzed. What capabilities already existed that could be enhanced to provide more effective training? What new capabilities needed to be developed? How should these be presented to the aircrews? From the onset, with regulator support, it was agreed that the AAMP training should be conducted as "non-jeopardy" events similar to the windshear training program. This would prevent the training from becoming a graded event where the pilot has to meet some criteria for completion.

Numerous questions arose as to the effects of implementing the training events. Would the equations of motion for the aircraft simulation and the motion system support the extreme attitudes? Would the dynamics of the maneuvers damage the simulator, particularly the visual system?

A review of aircraft incidents and accidents over a ten year period revealed that training related problems generally arise from three specific areas: meteorological conditions, unusual attitudes, and aircrew distractions. All simulators used for American Airlines pilot training are equipped for microburst/windshear training in accordance with FAA regulations. Therefore no modification of the simulators was required. However, to provide standardized training the desired shear intensity for both a takeoff and landing event were determined for each simulator type and these were incorporated on the AAMP Training Event Page of the Instructor Facility (IF). Aircrew distraction events consisting of an uncommanded stick shaker and rising terrain escape maneuver using the Ground Proximity Warning System (GPWS) were already installed. However, these were enhanced to better match the aircraft type of the simulator.

The AAMP events are controlled from a single page on the instructor facility. A sample page is shown at Figure 1. To simplify the setup each event is selected and activated from a single ICON. When the ICON is selected the simulator first sets up a "Parameter Plot" page on the alternate IF. The event then activates and selected parameters are displayed for student and instructor review. Should the instructor so desire a hard copy of the results can be sent to the printer. Early in the development of the program it became apparent that activation limits were needed on some of the events. The logic for the ICONS is such that they cannot be selected if the simulator is not within the allowable window.

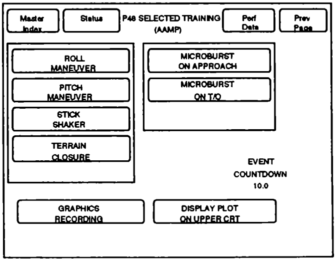


Fig. 1 - Typical Instructor Facility Page

The historical approach to unusual attitude training was for the student to shut his eyes while the instructor flew the aircraft into an attitude from which the student would recover. For years this has been standard practice for all military pilots flying training/fighter aircraft. However, with the increasing percentage of civilian only pilots in the airline industry many have never seen extreme unusual attitudes. In addition, former military pilots may not have seen unusual attitude recoveries for many years. Two maneuvers were developed: a rolling maneuver, and a pitch maneuver.

The Roll Maneuver

The rolling maneuver is designed to place the pilot in an inverted attitude from which recovery is possible if proper technique is employed. The event simulates a severe wake turbulence encounter. To attain the desired attitude several approaches were considered. The original idea was to overlay a "rotating wind component" on the ILS glideslope. This technique was abandoned as the results would not consistently give the desired learning objective. The other option was to either drive roll position or to induce a rate of roll that the pilot could not overcome. These could be activated anywhere in the flight envelope. To keep the pilot-in-the-loop the rate-of-roll technique was used. The event is enabled any time the aircraft is above 50 feet AGL and the airspeed is below 220 KIAS. When the event begins random number generators determine both roll direction and event severity. Moderate turbulence is activated and a slow roll rate to approximately 10-15° of bank begins. During this portion of the maneuver the pilot doesn't realize that he is not in control of the simulator and tries to correct with opposite aileron. The roll rate then reverses and ramps up to it's maximum value. The induced rate then ramps out so as to be completely out with the simulator in approximately 135-180° of bank. During the entire process the aerodynamics of the basic simulation are not changed. The maximum airspeed limit was required because under certain conditions the simulator could be "trapped" with the event activated if the student aggressively applied recovery controls and the aileron effectiveness equaled the induced roll. As an additional safety measure the rate input will terminate five seconds after start of the roll reversal. To give consistent results at various altitudes and airspeeds, the input rate is normalized by dividing by dynamic pressure.

The Pitch Maneuver

The pitch maneuver is designed to teach the pilot to take whatever action is necessary to prevent excessive nose high attitudes and thereby preserve the maximum airspeed. The event is activated the same way as the roll maneuver. With an induced pitch rate, the attitude is driven to a maximum of 60-70° of nose high pitch before releasing the simulator. The maximum pitch angle is linearly reduced as a function of bank angle such that with 60° of bank the maximum pitch angle before release is about 30°. The induced pitch rate is such that the aircrew cannot stop the attitude from increasing to maximum without executing the roll. Initially the event was executed without changing the aircraft longitudinal trim. After putting the event in training, it soon became apparent that for realistic training effects longitudinal mis-trim was needed. Without trimming if the pilot was slow to input a roll to help stop the pitch, and merely pushed on the column, the driver would eventually release and the simulator would recover to a Phugoid mode on it's own. The simulation was modified so that the event now starts with the autopilot engaged. When the event is activated the runaway nose up stabilator trim malfunction starts driving, followed closely by the pitch up. As a result the simulator will not recover on its own. One drawback of this technique is that the pilot gets an indication that something is about to happen. However, before the pilot can react to stop the trim the simulator is into the pitchup from which he must recover. The modification was successful in all simulator types except the Boeing 727. The design of the stabilator trim system control loader mechanism in the Level C/D 727s simulators is such that considerable damage occurred when the crew attempted to stop the runaway trim with the autopilot engaged. A revised stab trim brake system is being developed to eliminate the problem.

The GPWS Escape Maneuver

A fully operational (GPWS) was already integrated in all simulators. To induce a terrain warning in the simulator a feature was added that causes the host computer to increase the local terrain elevation. The rate of climb of the local terrain was tailored for each simulator type so that at training weights, during a properly executed escape maneuver, the simulator will out climb the rising terrain. The terrain will continue to climb until the event is terminated by the instructor. The terrain will then descend back to local elevation at approximately half the rate used in the climb. This gives the crew the illusion of having crested a mountain ridge.

The Uncommanded Stick Shaker

The uncommanded stick shaker induces cockpit chaos into the simulator. When the event is activated the stick shaker on either the Captain's or First Officer's control column becomes active for no apparent reason. Implementation of this event was relatively easy in all aircraft types except the Boeing 757/767. In these simulators when the stick shaker is activated without encountering stall angles of attack the EICAS (Engine Indication and Crew Alerting System) initially would give erroneous messages. The problem was eliminated by stimulating the EICAS system with an artificial angle of attack which drives the Pitch Limit Indicators (PLI) on the attitude indicator to the aircraft symbol.

LESSONS LEARNED

American has been using the unusual attitude training maneuvers for approximately six months with excellent response from those undergoing the training. To date no adverse affects on the simulator or visual systems have surfaced. Initial concerns that the violence of the maneuvers would excite a harmonic in the wide projection systems with the potential for cracking the mirrors have not materialized. No modifications of the motion response was required for the newer generation simulators. However, the pitch up maneuver resulted in persistent motion tracking errors on the older Link Advanced Simulator Technology (AST) simulators. These errors were eliminated by a twenty percent reduction in the motion response while the event is active. The reduction is ramped in and out so that it is transparent to the aircrews.

Initially the equations of motion for the DC-10 simulators appeared to limit the roll attitude to approximately 90° of bank. If the roll maneuver was flown with the landing gear extended it would "crash" before releasing control to the aircrew. However, with the landing gear retracted the maneuver worked as planned. The problem was finally traced to the simplified numerical algorithm method used on the older generation simulators for calculating the tangent of the bank angle. The calculation resulted in the length of the landing gear appearing to go to infinity as the bank angle approached 90°. The problem was eliminated by limiting the length of the landing gear to it's maximum length with the struts extended.

Points of Contact

Ronald W. Shoulars
Manager, Simulator Evaluation
American Airlines Flight Academy
P.O. Box 619617 / MD 876
DFW Apt., TX 75261-9617
Phone (817) 967-5523
FAX (817) 967-5127
email: Ron_Shoulars@amrcorp.com

Asok Ghoshal
Manager, Simulator Engineering
American Airlines Flight Academy
P.O. Box 619617 / MD 872
DFW Apt., TX 75261-9617
Phone (817) 967-5572
FAX (817) 967-5057
email: Asok_Ghoshal@amrcorp.com

AIAA-96-3488-CP

Quality Flight Simulation Cueing
Why?

Paul A. Ray
Manager, National Simulator Program
Federal Aviation Administration

Abstract

The issue of flight simulation cueing is the subject of many proposals from highly motivated individuals and organizations. Such proposals take on many forms. Some would advocate the removal of motion from all simulation requirements. Others would propose that "PC" based representations of systems are sufficient to qualify as a Flight Training Device for a specific airplane.

This paper addresses the causes for the need for accurate cueing in simulation and the potential impact of less than faithful replication of those cues. Specific areas addressed include motion, visual and tactile feel cueing requirements. Examples of the results of less than accurate cueing are provided. Experiences of simulation within the medical community are reviewed and addressed regarding the impact of less than realistic simulation cueing.

Introduction

Airplane flight simulation devices are defined by the Federal Aviation Administration (FAA) as flight simulators and flight training devices (FTD) meeting the standards of Advisory Circular (AC) 120-40B, Airplane Simulator Qualification and AC 120-45A, Airplane Flight Training Device Qualification. The ultimate use of either device is the qualification and certification of flight crewmembers to operate the aircraft simulated in commercial operation without the necessity to demonstrate proficiency in the same event(s) in the actual aircraft. The levels of simulation devices described by the FAA are founded upon the realistic replication of the aircraft flight deck, systems, performance, handling qualities and environment in which the specific simulator or FTD is designed to provide assessment of flight crewmember performance. The most readily noticeable difference between FAA flight simulators and FTDs is the consistent combination of motion and visual systems on flight simulators and the absense of a required motion or visual system on an FTD.

If a crewmember is to be assessed in a ground based vehicle, in lieu of the actual aircraft, a regulatory agency charged with upholding the highest standards of aviation safety and maintaining the publics trust in the safety of the aviation industry must insist upon simulation which replicates the aircraft.

To obtain the behavioral performance that must be assessed, as it would be assessed in actual flight, crewmembers must be stimulated in simulation to perform as they would in the aircraft. If the simulation is to properly stimulate the physiology of the human being to achieve performance as it would occur in flight, it is very clear that simulation must faithfully address the cues upon which that flight performance is based.

Frequently the training and simulation community, and yes, the regulatory community too, honestly attempt to dissect a maneuver or procedure out of actual flight and subdivide it into its smallest components through simulation. If we are not extremely careful we might lose sight of what actually occurs inflight with the accompanying cues upon which the pilot or crewmember formulates their decision making process and resultant behavior.

The human being is continually exposed to a vast amount of "data" which we assimilate and make conscious and unconscious decisions. Data, as presented to a pilot or crewmember, comes in various forms and results in a decision to either maintain or change system(s) operation or an aircraft condition. That data can be summarized, or classified within the basic sensory receptors of the human being. Typically, in flight simulation, we are dealing within the broad categories of sight, sound, and motion.

Foundation of The Cueing Issue

An aircraft, in any phase of ground or flight operation, provides every pilot or crewmember with the same cues. The aircraft, its performance and systems operation, always responds the same to a specific control input (systems or flight control) under the same given conditions.

Why then does the aircraft and it's systems perform differently under the same apparent circumstances? The obvious answer, and foundation for seemingly endless research in numerous fields of endeavor, is the human being. Pilots, contrary to a minority opinion, are also human beings.

Although exposed to the exact same set of cues, whether driving a car, a golf ball, or an airplane, performance differs between human beings. We are supposedly born with the same set of sensory receptors. However, the sensitivity of those receptors and more importantly, the integration of the information received is processed by the brain of what we are told are unique individuals.

If every human being processed identical information in an identical manner, not only would life be rather boring, our job in simulation would be simplified somewhat in that decision making would be an exact science.

Even if we incorrectly assume an exact decision making process, we are presented a major problem when we observe, and frequently record, what appears to be identical pilot performance to the same stimuli due to the varying acuity of the sensory receptors of different individuals. Those with a keen sense of sound and or sight may satisfactorily compensate for a loss in feel to produce the same performance as others. Similarly, does the pilot with a keen sense of feel use sound or sight to the same degree as those with less sensitive feel, or motion sensitivity? Most behavioral psychologists will confirm that humans incorporate, interact with, and respond to their environment in an infinite number of ways, thereby forming the centerpiece of focus for flight simulation.

Cueing Creates Behavior

What do we mean by "behavior?" Webster's definition is "...anything that an organism does involving action and response to stimulation, or the response of an individual, group, or species to its environment." To better understand the idea behind "behavior" we incorporate much of our introductory discussion to recognize that there are essentially three aspects covered when flight training or instruction is conducted -

- first, a cognitive understanding or knowledge of procedures (i.e., what is supposed to take place and why - what pieces of equipment are to be used and how, etc.);
- second, an understanding of the sequencing of events, (i.e., what follows what in a procedure - what are results of control input, etc.) and what actions should be taken to maintain or to correct the situation.; and
- third, a level of skill that will allow the student to manipulate the controls to achieve and maintain a desired attitude, altitude, airspeed, location, or a combination of these factors.

When exposed to flight tasks, each student develops at least some of each of the above areas - procedures, sequence of events, and skill. When these factors are taken collectively and combined with the information processing and decision making conduct of each student (*i.e.*, acknowledging or responding to some collection of stimuli provided by the environment) the result is the development of behavior. If this behavior was exhibited over a long enough period of time and the results of the processes were deemed to be satisfactory, that learned behavior is reinforced. However, it is critically important to remember that acceptable pilot behavior is pilot response to stimulation from his environment, stimulation as perceived in his cockpit. But his response is made up of a combination of ***all*** of the following:

1. knowledge of equipment;
2. understanding and ability to apply knowledge of procedures;
3. decision making process involving control application strategies; ***and***
4. skill in manipulating the controls in accordance with those strategies to achieve and maintain aircraft attitude, altitude, airspeed, and position.

Most expert flight instructors and educators would agree that piloting tasks, in general terms, could be described as a collection of cognitive and psycho-motor tasks. In the aviation training business, we hear these terms used rather routinely. A review of these terms and their definition provides us with a focus on the application of quality flight simulation for training and evaluation. From Webster's New Collegiate Dictionary:

- **psycho-motor** = of or relating to motor action directly proceeding from mental activity.
- **cognitive** = of, relating to, or involving cognition.
- **cognition** = the act or process of knowing, including both awareness and judgment.

From our familiarity with a flight simulator, we know that it certainly can be used to teach or train a cognitive task, and that its use does, indeed, add to a student's cognitive understanding. However, a simulator probably lends itself more appropriately or more completely to a reinforcement of what is already cognitively understood - a "mission" more logically immersed in the teaching or training of psycho-motor tasks. Then, in order to determine how technically complex a flight simulator must be, we must know what is necessary to teach or to train psycho-motor activities in that simulator. Again, our colleagues in the education field have regularly said that learning a psycho-motor activity involves four steps - plus one more (a fifth), if warranted. They, and their definitions, again, according to Webster, are as follows:

1. **demonstrate** = To show clearly. To illustrate and explain, especially with many examples. To show or prove the value or efficiency;
2. **attempt** = To make an effort to do, accomplish, solve, or effect;
3. **correct** = To make or set right. To alter or adjust so as to bring to some standard or required condition;
4. **practice** = To perform or work at repeatedly so as to become proficient. To train by repeated exercises; to do repeated exercise for proficiency. "Proficient" means having or manifesting the knowledge and experience needed for success in a trade or profession. [Shared meaning with synonyms of Proficient: *Adept, Skilled, Skillful, Expert.*] and, if warranted;
5. **examine** = To inspect closely. To test the condition of. An Examination is an exercise designed to examine progress or test qualification, proficiency, or knowledge.

We believe that the educational specialists are correct. In fact, many of us, in our own background, have anecdotal experiences to verify that these are, indeed, the steps involved in teaching or training a psycho-motor activity - with "practice" as perhaps the most significant of the group.

Application of learned behavior in an aircraft requires the student to go through the decision making process we discussd earlier. The student must decide what and how to apply learned experiences to the new set of circumstances; i.e., what behavior is proper within the environment - the aircraft cockpit. In order to assess proper transfer of behavior, a particular device must accurately replicate the aircraft, its functions and flight characteristics. Our goal must be to have skills learned in simulation, if used to replace actual flight training, transfer directly, without modification, to the aircraft. The term we use to describe accurate simulation is "bi-directional transfer of behavior." If a pilot must alter the cognitive and/or psycho-motor skills developed in flight simulation when transitioning to the aircraft or the cognitive and/or psycho-motor skills developed in the aircraft require any modification in flight simulation, the proper transfer of behavior has not occurred.

The Power of Realistic Cueing

The application of simulation technology is ever expanding. Many are familiar with simulation applications within the commercial and military aviation community, including space flight simulation. Perhaps lesser known, yet very powerful applications are continuing in such diverse areas as super tanker shipping, automotive driving, combat tank driving, fire-fighting, and nuclear powerplant simulation.

Another area which may appear to have no application to flight simulation is the area of anesthesiology simulation. However, the approaches taken to anesthesiology simulation should not surprise members of the aviation training community. Similarly, the results of their efforts should not be unexpected to knowledgeable simulation users. Comments regarding their experiences with simulation are certainly relevant to our discussion of flight simulation cueing.

One group of anesthesiologists has used a "PC" simulation with a desktop computer running a sophisticated model of human physiology and pharmacology to determine accurate and precise responses to operating room procedures, using a monitor and "mouse" to perform the actions of an anesthesiologist. ANESTHESIOLOGY, Vol. 74, No. 4, April, 1992, pg. 495, contained the following comments in "Anesthesiologists' Management of Simulated Critical Incidents", by Howard A. Schwid, M.D., and Daniel O'Donnell, Ph.D.:

> "... an anesthesia simulator was used to evaluate the management of simulated emergency situations by ten anesthesia residents, ten facility anesthesiologists, and ten anesthesiologists in private practice in order to identify specific patterns of errors in diagnosis and treatment. The simulator is a computer program that presents the patient, monitors, and management choices in a graphical display on an IBM or compatible personal computer. Many errors were observed in the management of these emergency situations, and even anesthesiologists with years of experience made serious errors. ...Only 40% of subjects correctly diagnosed simulated anaphylactic reaction; 27% adequately treated simulated myocardial ischemia; and 30% managed a simulated cardiac arrest according to Advanced Cardiac Life Support (ACLS) guidelines. Problems with continuous infusions of vasoactive agents were common. Fixation errors or failure to revise a plan in the presence of inconsistent cues were made by 63% of subjects."

ANESTHESIOLOGY, Vol. 74, No. 4, April, 1992, pg. 491, "Improving Anesthesiologists' Performance by Simulating Reality" contained the following remarks regarding the simulation efforts reported by Schwid and O'Donnell:

> "...The results were sobering. ...A few caveats about the study are in order. How does case management using this simulator differ from the "real world"? Schwid and O'Donnell acknowledge that the computer-screen based simulation is *not* the real operating room (OR), and so it is impossible to know whether these subjects would have performed as poorly in real life as in this study. Working "in" the computer screen OR is considerable different than working in the real OR, and regardless of how facile the subjects became with the simulator, its artificial nature might well have adversely affected their performance, especially regarding rapid dynamic responses that may rely on subtle environmental cues to trigger and guide them. the artificial environment can be a two-edged sword. Some activities are

> probably easier to perform on the computer screen..., whereas others are actually easier in the real OR than on the computer screen. In addition, this simulator does not reproduce the human-machine interaction problems that plague real case management, nor does it duplicate the psychological and environment of the OR in which communication, teamwork, and management of distractions are important issues. ...The high frequency of fixation errors in this study, even among experienced practitioners, is of some concern, but is not unexpected. ...These kinds of failures of interactive diagnosis and response are probably due to fundamental aspects of human cognitive psychology in complex and dynamic work environments. Fixation errors appear to occur because of limitations in our ability to deal with dynamic situations in which the available cues incompletely identify the causative feature, forcing us to make an initial guess as to what is occurring. ...The persistence of the fixation is also related to human tendencies to make data conform to preexisting expectations, and it is exacerbated when relevant data are hard to acquire or are masked by other data and tasks."

Another group implemented a simulation which appears to replicate an operating room, complete with actual operating room monitors and equipment. ANESTHESIOLOGY, Vol. 69, No. 3, September, 1988, pg. 388, "A Comprehensive Anesthesia Simulation Environment: Re-creating the Operating Room for Research and Training", by David M. Gaba, M.D. and Abe DeAnda, B.S., contained the following description of the "operating room" environment approach:

> "...Added realism is provided by requiring that monitoring and other tasks be performed using standard operating room (OR) equipment wherever possible. Furthermore, commercially available "simulator" are used to provide waveforms and data to patient monitoring equipment.
>
> ...A second essential characteristic of a comprehensive simulator is that it re-creates the anesthesiologist's physical, as well as mental, task environment. Simulation takes place in a real OR, and the subject is required to physically perform tasks wherever appropriate, including airway and tracheal tube manipulation, physical examination, drug selection and administration, and equipment trouble-shooting. This heightens the realism of the simulation and forces the subject to reallocate time and attention when physically performing a task. Interaction with the "surgeon" is important both for acquisition of clinical information (which may be correct or misleading), and also because it is potentially distracting. The role of surgeon, and that of circulating nurse, are played by the simulation director."

Results of the operating room environment effort have received very positive results. Evaluation of the simulation realism ranged, on a scale of 1-10, from 8.1 to 9.3 for case presentation, anesthesia equipment, instrument readings, responses to drug administration and physiologic responses of the simulator, and simulated incidents (abnormals, emergencies). Not surprisingly to the developers, the mannequin used to simulate the patient received significantly lower 'realism' assessments due to overall appearance, airway, and responses (*i.e.*, breath sounds)

The anesthesiologist's experience with simulation should not surprise objective observers in the flight simulation arena. As will be discussed later, the use of "PC" devices as an educational tool in the introductory phases of flight training have probably be underutilized.

"PC Simulation"

As noted by the anesthesiologists, PC devices which lack the ability to provide sensory cueing as would be experienced in the real world create major behavioral (*i.e.*, performance) problems if used as an assessment tool for real world performance. However, PC based training should not be summarily dismissed. It can be used as a highly beneficial training tool. Quality applications of desktop computer capabilities clearly provide a highly efficient vehicle for the standardized introduction and review of a wide range of material (*i.e.*, systems operation, regulations, procedural/explanatory information on aircraft and flight procedures). Not only should a PC be capable of improving the standardization of information, proper applications should greatly improve the efficiency of most educational processes. However, as a replacement for actual flight training or as an assessment tool, the results of the anesthesiologists application of such tools clearly supports the position that such devices are inappropriate for final behavior development or assessment.

The Power of Simulation

The need for 'realism' in simulation cannot be overstated if we are to use a simulation device to train and, particularly, evaluate behavior that must transfer correctly from, and to, the aircraft.

Much as the PC devices employed by the anesthesiologists did not provide the correct cues, a few individuals and companies strongly advocate the use of screen based devices as a replacement for a spatially and tactically correct cockpit environment.

In their paper,"THE USE OF GRAPHICS FOR INTEGRATED PROCEDURES TRAINING", presented at the May 15-16, 1996, Royal Aeronautical Society Flight Simulation Conference, Iain Wilson of Traning Systems Technology and Graham Ketley of Quadrant Systems suggested the training communtiy depart from the physical fidelity requirements of today's FTD and adopt "...touch screen technology coupled with powerful graphics packages..." The authors accurately noted the lack of tactile realism in their proposed acceptance of touch screen "simulation", driven by the reduced cost of such a device. Addressing the functional fidelity of such devices, Wilson and Ketley noted, "...In this contest Functional Fidelity, refers to the extent to which the device is capable of reproducing the behaviour of the equipment in response to the trainee input." We agreed that the behavior of the equipment response must be correct. We must likewise be assured the pilot behavioral response is correct. Without the proper cueing, tactile feel and spatial orientation in the instant case, we cannot be assured of behavioral accuracy.

As with the previous comments on PC simulation, devices such as proposed by Wilson and Ketley should not be dismissed from consideration for inclusion in a comprehensive training program. Such devices, properly developed and integrated into a quality training program could contribute to improved efficiency.

The power of simulation to create and modify behavior is an obvious theme in our look at simulation cures. FAA insistance upon the correct "fit, form, and function" which includes the correct spatial orientation and tactile feel within the cockpit is an outgrowth of the power of simulation. At least three instances come to mind which reinforce the requirement for correct cueing in simulation. Some may be aware of at least two incidents which resulted in landings of aircraft with the landing gear retracted wherein the crewmember did not exert sufficient force to move the emergency gear handle "over center" and release the gear off the uplocks. In both cases the aircraft was raised, placed on jacks and maintenance extended the landing gear with the emergency gear extension handle. When questioned, both aircrews indicated they had supplied at least as much force as they had applied in the simulator and thought the aircraft gear to be frozen in the 'up' position. Checking the simulators in question,, the numerous emergency extensions practiced, had worn the cabling and pulleys to the point where the simulator forces required were significantly less than needed in the aircraft. Another more recent incident involved a pilot's attempt to shut down an engine with a fire handle. He thought a compound emergency had been encountered when, applying at least the same forces as performed in the simulator, the engine would not shut down. Following an uneventful landing, maintenance found the fire handle in normal operating condition. As with the emergency landing gear extension problems, forces required on the simulator fire handle had slowly decreased following multiple activations during training and checking.

Additional testimony to the power and effectiveness of accurate simulation cueing was shared at the May 15-16, 1996, Royal Aeronautical Society Conference with the comments of an airline pilot who had, without warning and well clear of any thunderstorms, encountered a severe windshear during an appoach to landing. He was convinced he could not have survived the encounter without the required flight simulator windshear training.

The Hazards of Incomplete Cueing

In the past, well-meaning observers of the aviation training community have suggested that to obtain the best possible simulation device to perform flight crew member training or checking, one would only need obtain from pilot subject matter experts a listing of the "cues" necessary for given tasks, build a device to provide those cues, and assume they could grind out competently trained flight crew members. Unfortunately, for any given number of pilots asked to describe the set of cues used for a given task, the number of answers would be equal to a number larger than the number of pilots questioned. No two pilots use the same set of cues, to the same degree, for any task. Given an unlimited amount of time, most pilots would probably change their minds about what cues or how much of any cue they really do use, assuming they could quantify the cues. Structuring any kind of "simulator" with this kind of input is, at best, a non-conclusive form of simulation.

As long ago as 1966, in his publication, "Philosophy of Simulation in a Man-Machine Space Mission System" for the US National Aeronautics and Space Administration, Mr. T. M. Fraser cautioned that "...a situation, task, or problem can be simulated in a valid manner only insofar as its parameters are known" and, perhaps more importantly, said "...simulation may provide an incorrect solution or false information if the simulation is incomplete or the parameters of the simulation are incorrect." In recognizing these challenging situations, John M. Rolfe, Senior Principal Psychologist with the United Kingdom's Ministry of Defense, together with colleague Ken J. Staples of the Royal Aircraft Establishment in Bedford, U.K., in their book "Flight Simulation" (Cambridge University Press, 1986) has provided a major clue in the resolution of this question with the following comment:

> "There is in fact considerable evidence that the interpretation of a given set of limited information varies widely between individuals. And it is limited information with which we are concerned...[In a simulator] we can produce only a very limited subset of the information available in the real world but this subset, while potentially useful to some individuals, may not be appropriate to others. Fortunately the human being is very adaptable, and the world is full of redundant information. So if preferred information is absent then some other relevant, though perhaps less easily interpreted, information will be selected."

So then, if we are not going to use aircraft for training or checking, what alternative exists? If the answer is simulation, we must give the pilot trainee the greatest possible source of cueing information, through which he may sift, sort, and select those cues that can and will be used to accomplish a task. This cueing information must be presented as it would occur in an airborne aircraft: i.e., the systems and equipment used to present these cues must be presented in the same manner - in the same "form, fit, and function" - as the pilot would receive, and react to, in the aircraft. The critical aspect of which we must be aware here, is the behavior that is being trained, modified, and/or reinforced - *i.e.*, the behavior that will be acceptable when exhibited in the aircraft.

However, we must pause to be sure we understand a very critical aspect of simulation. Simulation that merely *appears* to provide practice of the desired skills, in the attempt to re-inforce behavior that is similar to the desired behavior, is to misunderstand and misuse the function of simulation - and may well introduce unwanted and improper behavior. The behavior learned and reinforced through simulation is behavior that is relevant only to that simulation device and only under the same circumstances. Merely accomplishing a task in a simulator, whether or not it is accomplished to some level of "proficiency," is not, in itself, evidence that the simulation is or should be acceptable in terms of operating the aircraft in the real world. To reiterate, more than just skills transfer - all decision making and control application strategies, together with the necessary skills and the prerequisite knowledge of procedures and an understanding of the sequence of events - all transfer. We must be sure that the behavior that transfers is behavior we desire to have exhibited in the aircraft. Attempts to utilize incomplete cues in simulation will certainly alter behavior which would be exhibited in the aircraft, thereby jeopardizing the safety record we have contributed to with quality flight simulation.

Visual, Sound, and Motion Cueing

Setting aside form, fit, and function issues; sound, visual and motion cueing typically occupy the majority of time devoted to flight simulation discussions.

Rapid advancements in computer technology are most evident in today's flight simulator visual systems. The vast impovements in realism, and thereby more effective cueing, provided by state of the art systems has been driven primarily by pilot demands for improved cueing. The majority of new systems are coupled with a "wide" display system of at least 150^0 continuous field of view (FOV) Several recent systems have been put into training with more than 200^0 FOV (*i.e.*, Boeing Aircraft Company B-777 flight simulator).

The importance of visual cueing and the improvements to realism were amplified at the May 22, 1996 training seminar hosted by the Regional Airline Association (RAA) in Orlando, Florida. Mr. Doug Myers called for improved visual realism and increased FOV in flight simulation for RAA member airlines.

The addition of a visual system to a fixed base FTD can provide exceptional increases to the perceived realism of the device. The visual reinforcement of successfully 'flying' an approach and actually seeing the runway in front of you at minimums is a very powerful tool. In 1995, Delta Airlines, in cooperation with IVEX Visual Co. and Atlantis Aerospace, added a single channel visual to an FTD and received very positive comments from the crews who used it for training. However, some have mistakenly theorized the addition of a visual system alone is sufficient to create the cues needed for additional flight maneuver assessment. The following comments should address why such a conclusion would be erroneous.

In addition to the realism of the 'outside' world in simulation, visual systems also provide a strong perception of motion. As Dr. Edward A. Martin noted at the AIAA 12th Annual Flight & Ground Simulation Update, January 1996, "The principal perceptual systems of concern in motion and force simulation are the visual, vestibular, and haptic or tactual..." Martin added, "...Experiments regarding the interaction of the visual and vestibular systems have shown that when visually observed motion is reinforced by vestibular stimulation, there is a rapid onset of vection. In the absence of vestibular reinforcement, however, the onset of vection requires some <u>five to ten seconds</u>. Further, a conflict of vestibular information with visual motion results in a precipitous loss in vection..." Dr. Martin may have also pointed to the reason pilots are asking for greater FOV when he pointed out that peripheral vision creates a more pronounced sense of motion than central vision which is the visual stimulus area of the classical monitor based, particularly single channel, visual systems.

The cue conflict created by visual motion, unsupported by vestibular stimulation, is the likely cause of 'simulator sickness' or nausea fequently experienced by use of a wide FOV without motion. My first encounter with this phenomanon was in a visual system manufacturer's facility in 1988 during a demonstration of a richly detailed visual scene on a 150^{0} FOV display. The onset of nausea was nearly instantaneous with the perceived motion. Similarly, poorly synchronized (visual movement before motion) visual and motion sysems can also cause nausea or 'simulator sickness'.

Sound cueing is also a major player in realistic simulation. Flight without sound may only occur in space. Soaring advocates have noted their perceived increase in hearing sensitivity inflight which appears to be an aid in judging speed. In an airplane many pilots rely heavily upon engine sound during flight, particularly during an instrument approach. The knowledge of engine sound and resulting approximate power settings affords the opportunity to reduce the scanning of engine instrument and increasing the scan of flight performance instruments resulting a potential improvement in the accuracy of an instrument approach.

Accurate sound, particularly in turbo-propellor driven aircraft, may be more important than we realize. The high volumn, low frequency sound created by turbo-propellor aircraft can, and frequently does create vibrations sensed by the haptic, or tactual system described by Dr. Martin. The absense of such cueing in flight simulation causes accomodation, or alteration of behavior, by the pilot.

Simulation of the motion cues that occur inflight can, and frequently does, produce some of the most animated discussions within the training and simulation community. The issue, and the insuring discussions, tends to become more objective when participants segregate the motion cueing requirements for the tactical (high G force) environment of a military fighter aircraft from the operation of any aircraft within a normal training or checking environment (non-acrobatic).

A very large percentage of motion research has been devoted to motion requirements for military fighter aircraft. Frequently the data collected during fighter motion research and the resulting conclusions and recommedations for motion requirements, including tactical high-g flight, are incorrectly applied to all discussions and application of motion. It may be impossible, without the use a centrifuge type 'motion' system to accurately provide continuaous and realistic motion cueing for tactical fighter simulation. Although, as Dr. Frank M. Cardullo pointed out at the 12th Annual AIAA Simulation Udate, there are serveral systems available to augment fighter (acrobatic) motion simulation, including G-Seats, Vibration Systems and High-G Augmentation Devices. Dr. Cardullo's comment "It seems reasonable to assume that motion requirements may be dependent on the extant motion environment" brings us back to focus on motion simulation for the 'typical' non-acrobatic training environment.

Cardullo notes that flight motion has been devided into two basic categories: disturbance motion and maneuver motion.

> "...Disturbance motion is characterized by its stochastic nature, and is due to external, usually atmospheric, conditions. It may be either random continuous or random discrete. The random continous disturbances are things like buffet and 'rough air'. Whereas random discrete disturbances include phenomena such as wind shear, wind gusts, and atmospheric pressure discontinuities.
>
> Maneuver motion has three constituents; pilot initiated discrete changes in flight path, pilot continuous control input requred for precision tracking in high gain closed loop control tasks and pilot continuous control of a vehicle with low satbility. ...Wind shear, which was previously described as discrete disturbance motion, is in reality a combination of discrete disturbance motion when the shear is encountered, and maneuver motion in controlling the aircraft through the disturbance. ...In some cases, the onset of a phenomenon can be considered to be in the disturbance category, but the recovery from it would be maneuver motion and usually of the high gain, closed loop type. Examples of this would include failure

> modes and the previously mentioned wind shear encounter." Additionally, "...According to Hall (1989), motion cues are more important for high workload, higher gain maneuvers, especially with poorer visual cues, necessary for high gain tasks even with good visual cues and..."

Cardullo addresses the issue of simulator performancel assessment with the following:

> "It should be noted at this point that the appropriate metric applied to training simulation has been transfer of training, whereas the metric applied to R&D simulators has been performance. There are some who believe that performance metrics are appropriate in both cases, particularly since it is extremely difficult to conduct valid transfer of training studies. However, there is the countervailing point of view that good performance in the simulator is not indicative of a high level of transfer of training. However, this may depend on the definition of 'good performance in the simulator'. Or perhaps what should be measured is behavior. That is, if the simulator elicits behavior that corresponds to desired behavior in the airplane, then the simulation is said to be appropriate or correct."

Amplifying Cardullo's comments, we would perform an extreme disservice to the customers of simulation if we assumed satisfactory performance by a pilot in a consciously incomplete simulation was acceptable. In doing so we would thereby include some pilots whose perfomance could be unacceptable in the aircraft and exclude others whose unacceptable simulator performance could otherwise be acceptable in the aircraft.

In summary, today's high quality motion systems possesses system delays of 50 msec or less and, with a 6 degree of freedom (DOF) system, provide for highly realistic pitch, roll, yaw, heave, sway, and surge motion cueing to the simulator cab, much as the floor of an aircraft transmits the resultant motion inflight to the pilot. cueing for the training and evaluation of pilots thoughout the typical training environment. As noted by Martin and Cardullo, quality motion provides the critical integration of cueing to avoid cue mismatch within the pilot.

Ian W. Strachan, in his May 17, 1995, "Cueing For Motion" presentation to the Royal Aeronautical Society summed up my thoughts on motion cueing when he provided the following: "The UK Institute of Aviation Medicine (IAM) have agreed that motion platforms are the only simulation devices capable of fully stimulating the pilot's vestibular apparatus and imparting realistic accelerations in all of the 6 DoF to the whole body, thus exercising the motion feedback-loop that pilots are used to experiencing in aircraft and without which a cue-mismatch will be detected by the brain."

Summary

The March 1996 issue on Aerospace magazine documented FAA Administrator David Hinson's comments which referred to flight simulation as one of the major reasons air safety have so dramatically improved over the past 30 years. Hinson noted 1960s predictions which had forecast more than 2,500 fatalities in air accidents each year by the mid 1990s. Flight simulation was pointed to by Mr. Hinson as one of the principal reasons we have not fulfilled those earlier predictions.

Entrusted by the public to uphold the highest standards of flying safety, the FAA would be abdicating that trust if quality simulation was not maintained and improved. Quality simulation used for the assessment of pilot skills can only be achieved through a faithful integration of cues in simulation as they would be presented to a pilot or crewmember inflight.

The FAA will continue to review the best possible sources of information and data regarding accurate simulation of the cues provided by the airplane. Innovative methods of implementation have, and will continue to be encouraged.

AIAA-96-3489-CP

AIR-TO-AIR MISSILE ENGAGEMENT ANALYSIS USING THE USAF TRAJECTORY ANALYSIS PROGRAM (TRAP)

Joseph W. Herrmann
Aerodynamic Weapon Design Branch (TANW)
National Air Intelligence Center
Wright Patterson Air Force Base, Ohio 45433-5648

ABSTRACT

The primary focus of this paper is to demonstrate the various techniques which can be used to characterize the kinematic performance of an Air-to-Air Missile (AAM) in aerial combat. The backbone of this analysis will be performed using the USAF TRajectory Analysis Program, or TRAP. TRAP enjoys a broad usage throughout the U.S. DoD and contractor communities, due in part to its extremely flexible approach to the design and analysis of air launched weapons. TRAP is uniquely designed to aid the analyst in a wide variety of missile performance problems, especially for a comparative or parametric study of air launched missile designs. The techniques presented to depict AAM kinematic performance, include static point missile analysis, missile flyout performance, single shot analysis, and Launch Acceptability Region (LAR) generation.

INTRODUCTION

TRAP is a versatile, general purpose aerodynamic weapon simulation which was developed and is maintained by the USAF National Air Intelligence Center (NAIC) at Wright Patterson AFB, Ohio. TRAP was originally designed by NAIC as an analytic tool to evaluate and predict the design and performance of foreign aerodynamic weapons in 1-v-1 engagement scenarios. It can model up to three vehicles -- a launch aircraft, a missile, and a target -- but is primarily designed to model and characterize weapons, including Air-to-Air Missiles, Air-to-Surface Missiles, Cruise Missiles and Unmanned Aerial Vehicles (UAVs). The TRAP models and the performance results they generate contribute to numerous threat products in support of the U.S. acquisition, operational, and technical intelligence communities.

TRAP is currently coded in FORTRAN and is structured to be extremely flexible and modular in both its input files and source code subroutines to facilitate customized model development and program execution. User friendly data files describe all aspects of the simulation including the engagement, guidance, autopilot, seeker, propulsion, mass properties, and aerodynamics. This flexibility provides a powerful tool for performing parametric trade studies on missile design options such as propulsion or guidance and control concepts. TRAP features modified point mass modeling for all vehicles through a complete 6 Degree of Freedom (6-DOF) modeling capability for the missile and its subsystems. TRAP's capabilities also include an AAM Launch Acceptability Region (LAR) generator, a constant *G* launch contour generator, and a missile flight reconstruction capability.[1]

This paper is primarily concerned with demonstrating the types of analysis associated with the kinematic performance of AAMs rather than examining the nuts and bolts of any particular missile model or flyout simulation. To facilitate this demonstration, an unclassified TRAP missile model was developed specifically for this paper and is referred to as *Missile X*. Missile X was designed around a sidewinder-like airframe with increased maneuverability, a larger motor, and a greater seeker gimbal angle. Missile X was then exercised using TRAP and other tools to characterize its kinematic performance.

STATIC POINT MISSILE ANALYSIS

The first area of analysis examines the missile at isolated instances in time at fixed flight conditions. Without performing a complete flyout, there are certain performance capabilities that can still be quantified. This type of analysis has been categorized as static point analysis. In performing static point analysis, a snapshot is taken of a missile's capabilities at a fixed instance in time. These snapshots can then be represented in ways that lend insight into a missile's kinematic performance. Some of the types of static point analysis are missile *G* availability and turnrate performance. First however, the concept of aerodynamic trim and how it applies to point mass modeling must be understood. Also, a simple discussion concerning the differences between 6-DOF and point mass models is presented.

Aerodynamic Trim and 6-DOF vs. Point Mass:

In performing this type of static analysis we will examine the missile in a state of aerodynamic trim. Aerodynamic trim is an important concept to understand for many of the models used in TRAP. All TRAP point mass missile modeling is accomplished by applying the concept of trim. The missile is considered "trimmed" if, at a given flight condition (Mach, center of gravity (c.g.) and angle-of-attack (α)), the deflected control surfaces (δ) produce a moment which negates all other moments acting on the missile airframe, so there is no effective missile body rotation. Point mass modeling does not use an autopilot and makes the basic assumption that the missile is always trimmed in both pitch and yaw.

In a TRAP point mass model, the chain of events follows a much different path than that of the real missile. For every time step, the guidance command is evaluated directly against the available aerodynamic trim capability of the missile. The missile is then rotated to this new angle-of-attack condition subjected to the constraints that its angle-of-attack rate did not exceed an input allowable maximum. For a symmetric missile at nonzero angle of attack, a nonzero normal force will then be acting on the missile. The control surface deflections required for the missile to be trimmed at this new state are then determined, which define the incremental change in axial force due to both α and δ. All forces acting on the missile are then summed and the missile state is updated. The missile body rotations will have already occurred earlier in the process when moving to the new angle-of-attack state. Although not explicitly cited, this process also applies to both the yaw and roll channels.

Since the position and rates of the aerodynamic angles in pitch, yaw, and roll are controlled, and since the incremental change in axial force due to α and δ are taken into consideration, TRAP defines this type of modeling as *modified* point mass modeling to distinguish it from a basic point mass model.

The flow of events in a 6-DOF missile model corresponds approximately to the flow of events in a real missile. In a simplistic sense, a 6-DOF missile responds to a seeker error through the use of an autopilot, by generating a guidance command which is transformed to a demanded control deflection. The deflected fin creates a moment which rotates the missile airframe producing angle-of-attack which generates forces on the missile. All forces and moments acting on the 6-DOF missile are then summed and the missile state is updated.

In general, control systems are designed to maintain the missile in these balanced, trimmed states. Therefore, even though 6-DOF simulations do not require a complete knowledge of trim properties, the trim analysis techniques still have application. The following two examples are based on static point analysis of a missile in a trimmed state.

Missile G Availability:

Missile *G* availability is a basic measure of a missile's ability to maneuver. It can be characterized as a function of flight condition by examining the maximum trimmed α at which the missile can successfully fly. In general, this also will correspond to the point of largest normal force or C_N attainable by the missile at that flight condition. Equation (1) is applied for a specific missile mass and center of gravity to produce a contour map of maximum aerodynamic *G*.

$$G_{MAX} = [C_{N,MAX}\cos(\alpha_{MAX}) - (C_{A0}+C_{A\delta})\sin(\alpha_{MAX})] * [(S*q)/(m*g)] \qquad (1)$$

where...

$C_{N,MAX}$ = Normal Force Coefficient at maximum pitch trim condition. $f(M,c.g.,\alpha,\delta)$

α_{MAX} = Maximum trim angle-of-attack for this condition $f(M,c.g.,\alpha,\delta)$

δ = Control Surface Deflection (deg)

C_{A0} = Missile axial force coefficient at zero α. $f(M,Alt)$

$C_{A\delta}$ = Increment on C_{A0} due to α and δ. $f(M,\alpha,\delta)$.

S = Missile body cross sectional area (m^2)

q = Dynamic Pressure (N/m^2)

m = Missile Mass (kg)

g = Gravitational Constant. (9.80665 m/s^{2})

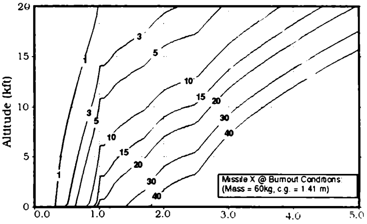


Figure 1: Maximum Aerodynamic *G*s Available

A contour map of the maximum aerodynamic *G*s available provides insight into a missile's maneuverability and possible operational flight envelope. ***Figure 1*** shows an example of a missile *G* availability plot for Missile X. This plot can be applied to a comparison of similar systems to determine where one missile has advantage and disadvantage in maneuvering capability. It can also be used to determine realistic flight termination conditions, since a minimum number of maneuvering *G*s is often a requirement against a specific target.[2]

Missile Turnrate Performance:

Another static measure of a missile's capabilities is a turnrate performance plot, also known as a "doghouse" plot due to its distinct shape. This type of analysis is normally applied to aircraft, but can also be quite useful when analyzing missiles, especially in comparison to aircraft or other competing missiles. ***Figure 2*** shows a doghouse plot of Missile X versus an F-15C. The isolines within each envelope are of delta longitudinal acceleration. Positive values indicate an excess power is available to accelerate. Negative values indicate that the vehicle will lose axial velocity at that condition.

From this plot we can see that Missile X has a distinct advantage in *G*s Available and turnrate performance through all regions of the flight envelope over an F-15C. Where the Missile X and F-15C limit lines converge (~Mach 0.8), indicates that you would not desire the missile's velocity to decrease below Mach 0.8 if you realistically expect to hit the target at this condition. In reality, the missile should always possess an excess in *G* capability over the target of 2 to 3 times the number of *G*s available to the target to ensure a successful intercept.

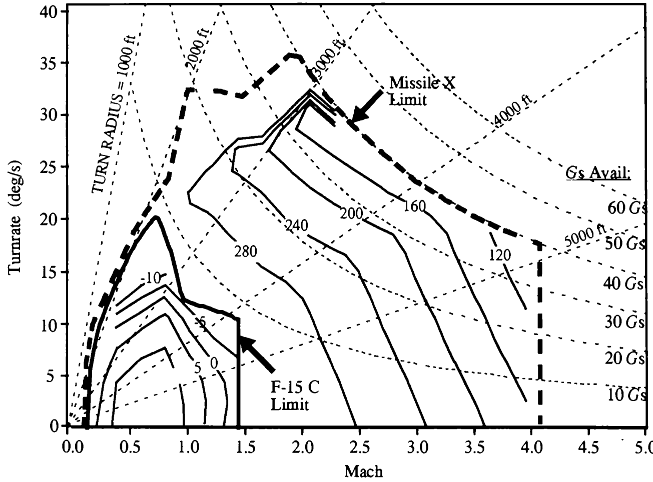
To perform the analysis for Figure 2, the missile and aircraft mass, propulsion and trim aerodynamic characteristics were evaluated at an isolated point in time. This information was processed using another NAIC product -- AIRCRAFT AVENGER, to produce the overlay shown in Figure 2. A family of these plots at various combinations of mass and c.g can fully describe the turnrate and longitudinal acceleration capabilities of a missile.[3]

MISSILE FLYOUT PERFORMANCE ANALYSIS

The next stage of analysis characterizes a missile's kinematic flyout performance in a target-less scenario. Examples of this type of analysis are constant altitude flyouts and constant *G* flyouts.

Constant Altitude Flyouts:

The constant altitude flyouts are a series of controlled, straight shots with the missile flying a



Conditions:

Missile X at mid-motor burn point (75 kg)

F-15C with 220 engine in maximum afterburner, no stores and 75% internal fuel

Altitude = 15,000ft

Isolines for Delta Longitudinal Acceleration (KCAS/sec) vs Turnrate and Mach

Figure 2: Doghouse Plot for Missile X vs F-15C

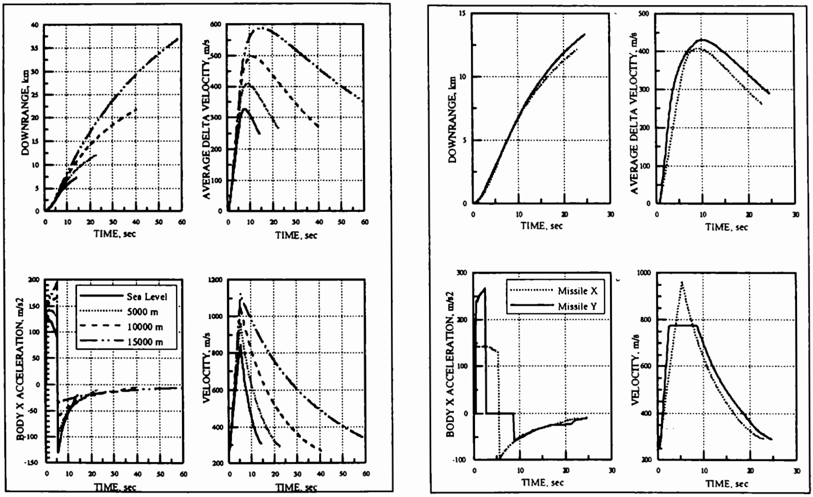
constant altitude guidance scheme in pitch. The launch Mach and altitude are varied through the expected operational envelope to characterize the non-maneuvering kinematic missile performance as a function of time. The common parameters of interest for constant altitude flyouts are missile downrange, velocity, and acceleration performance. ***Figure 3*** shows two constant altitude flyout sweeps plotted for Missile X. Along with the previously mentioned variables, we have also displayed the average delta velocity. The plot in ***Figure 3a*** demonstrates how Missile X behaves for a Mach 0.8 launch at various launch altitudes. ***Figure 3b*** shows how the same set of plots can be used to compare Missile X performance with a competing missile or possibly a similar threat missile at a fixed launch condition of Mach 0.8 and 5000m altitude.[4]

Constant *G* Flyouts:

A constant *G* flyout is very similar to a constant altitude flyout except the launch Mach and altitude are kept constant while a sweep is instead performed on the missile's pitch or yaw guidance command. ***Figure 4*** shows a constant *G* flyout for Missile X with a sweep in vertical guidance command from -5 *G*s to +5 *G*s in 1 *G* increments. Overlaid on this plot of missile flight path curves are the isolines for Mach, flight time, and missile *G*s available as functions of altitude and downrange. As did the constant altitude flyout plots, the constant *G* contours give the AAM analyst insight into a missile's kinematic performance. Superimposing constant *G* plots for different systems can aid the analyst in determining where areas of advantage and disadvantage exist. These tools can be valuable for the objective kinematic comparison of two missile design alternatives or could be used to compare competing threat and friendly systems.[5]

SINGLE MISSILE SHOT ANALYSIS

Once the missile is performing acceptably under these constrained shot conditions, a target is added to the scenario and more realistic single shots are evaluated. By a single shot, we mean that a solitary engagement is setup and executed in TRAP where the missile will fly out only once and the simulation will be terminated rather than continuing with another shot. ***Figure 5*** shows a simple example of how single shot analysis may be applied. It is a comparison between Missile X and Missile Y at 15,000 ft, Mach 0.8 from 70



3a) Altitude Sweep at Mach 0.8

3b) Missile X vs Missile Y, 5000m, Mach 0.8

Figure 3: Missile X Constant Altitude Flyouts

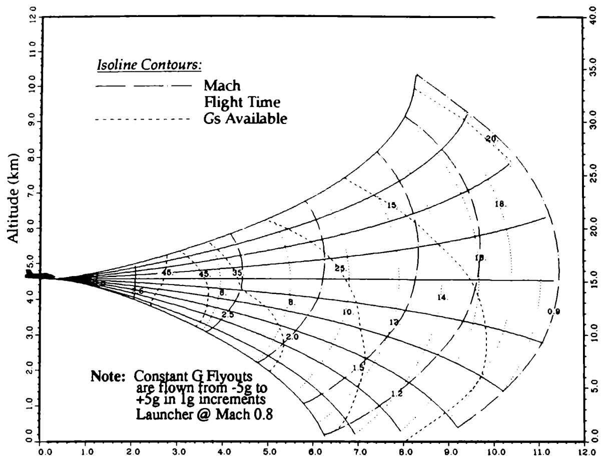


Figure 4: Missile X Constant *G* Flyout with Isolines

degrees off the tail. The purpose of this particular shot was to demonstrate the performance differences at minimum range between two competing missiles.

In this example, the result of the engagement for both missile's was a successful target impact, but if we had altered the scenario only slightly, moving the target closer to the shooter, we may have found the missile was no longer able to hit. There are numerous mechanisms of failure associated with AAMs in 1-v-1 engagement analysis. Understanding these "termination conditions" is critical to the effective analysis of threat missile performance.

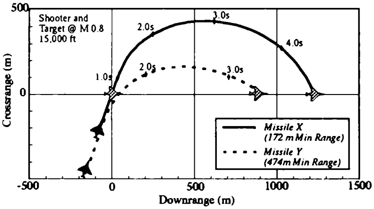


Figure 5: Single Shot Missile Comparison

Missile Termination Conditions:

When simulating an AAM launch, the selection of termination conditions will have a major influence on the reported outcome of a launch. Contrary to what their name implies, not all termination conditions are actually linked to the missile's flight ending. In many cases the termination conditions are user imposed values that will determine how the aircraft's fire control system will constrain the missile to be fired. Some of the more common termination conditions used in AAM engagement analysis are presented here. For discussion purposes, all termination conditions have been grouped into two categories -- *hard* and *soft* terminations. Hard terminations are those that physically end the missile flight, such as target impact. Soft termination conditions are those that are used primarily for launch aircraft fire control range determination or for the purpose of simplifying simulation. In contrast to hard terminations, soft terminations do not actually stop the missile's flight.

Maximum Guided Flight Time: The maximum powered or guided flight time of a missile is determined by the shortest lifetime of any of the critical systems on the missile. These include thermal batteries, pneumatic systems, cooling systems, and others. Once the first system power failure occurs, the missile will no longer function and often, shortly thereafter, will self-destruct. Other than the slight possibility of an unguided missile flying into a moving target, this is classified as a hard termination.

Minimum Flight Time: The minimum flight time of an AAM missile is linked with the minimum safe and arm (S&A) time of the fuze. In order to avoid the missile fuzing on the launch aircraft immediately after launch, an arming delay is incorporated which ensures the missile is well clear of the aircraft before arming the warhead. This is simply a safety measure and will not terminate the flight. If the missile reaches the target before the minimum S&A time, unless a direct target impact occurred, it would harmlessly pass, never fuzing. This by itself would be classed as a soft termination, however once the missile has passed the target, there are other termination conditions that would likely end the flight.

Broken Seeker Lock: There are numerous seeker related failures that can occur to terminate a missile flight. Some of these are associated with the physical or mechanical limits of the seeker head and its ability to track a moving target, and others are tied to signal processing and the ability of a seeker to "see" the target, discriminating it from its background. Additionally, the ability to process countermeasures and to discriminate between decoys can be a determinant to the success or failure of a missile launch. However, usually the seeker simply losing lock will not actually end the missile flight. In many cases, the missile will continue guiding based on the last known target flight path and in many cases, modern seekers will actively scan in hopes of reacquiring the lost target.

For catastrophic mechanical failures, such as gimbal lock, once the seeker gimbals are bouncing around uncontrollably, there is no hope of reacquiring the target, but the missile flight should still continue. From these arguments we can see that seeker related terminations should usually be modeled as soft terminations, but depending on the seeker's capabilities or lack thereof, can often be very hard terminations, depending on the mechanism of failure.

Target Impact: Target impact in TRAP is loosely defined as the missile passing the target's center of mass within the lethal radius of the missile's warhead. Kinematically, this means the missile had the energy and maneuverability to intercept the target. TRAP does not perform end game analysis. Whether the warhead fuzes or is able to inflict sufficient damage to destroy the target is not considered a factor in determining success. Target impact is a hard termination.

Minimum Missile Mach (or Velocity): This flyout simulation termination condition is probably the most misunderstood by TRAP users. Low missile Mach or low missile velocity is totally a fire control related parameter. When the fire control system determines the absolute maximum range of an AAM, or $R_{max}1$, it imposes a minimum allowable Mach number to ensure the missile will have sufficient energy at end game to destroy the target. This limit ensures that only shots with a relatively high probability of success are attempted. Increasing the low missile Mach used by the fire control algorithm decreases the displayed $R_{max}1$ to the pilot, but increases the energy of the missile at end game, thus increasing the likelihood of a successful shot.

Another major factor which can influence the selection of low missile Mach is the optimization of F-Pole. In a head-on engagement, with two aircraft firing Semi-Active Radar (SAR) guided missiles, F-Pole analysis is of particular interest. For head-on engagements, the low missile Mach is often set equal to the launch aircraft's speed at missile launch. By doing this, the launch aircraft will not begin closing on the target faster than the missile is closing on the target, thus maximizing the separation between shooter and target in a 1-v-1 engagement at missile impact.

Low missile Mach is a very soft termination because even if the missile has fallen below a certain limit, it will continue to fly and guide towards the target until a hard termination is satisfied. When analyzing weapons for which the fire control logic is unknown, selection of an appropriate value of low missile Mach should not be an arbitrary decision. Unlike many of the "hard" terminations, low missile Mach should be scrutinized carefully for each scenario and may possibly change for various user applications.

Minimum Closing Velocity: The minimum closing velocity of a missile is determined by the performance of the missile's proximity fuze. Most missile proximity fuzes operate by emitting energy (RF, laser, etc...) in a pattern around the missile in hopes of bouncing some of that energy off the nearby target. In order to detect the target, a minimum Doppler shift is required in the returned beam. If the missile's closing velocity, or overtake speed, with the target falls below a minimum value, the Doppler shift of the returning beam will be too small for the fuze to discriminate the target from its surroundings and the missile will fail to fuze. In this sense, minimum closing velocity is classified as a hard termination. However, an exception to this rule exists since the majority of AAMs also have backup impact fuzes. If a direct hit would occur before the fuze cone angle intersects the target aircraft, the missile warhead will still detonate. Therefore, low closing is a relatively hard termination with a limited exception.

Minimum *G*s: The minimum allowable *G*s can be interpreted as both a soft or hard termination. As a soft termination, minimum *G*s is related directly to low missile Mach since both are measures of a missile's available energy or maneuverability. Because of this relationship, minimum *G*s often are not used as a stand alone termination but rather their affect has already been taken into account when deciding the minimum missile Mach constraint. Minimum *G*s can also be a hard termination if the missile speed would actually fall below that necessary to sustain level flight (< 1 *G*). This is rarely an issue, but at extreme altitudes must be considered by the analyst.

Maximum (Minimum) Launch Range: A missile's maximum launch range is a soft termination and is related directly to the maximum value of $R_{max}1$ displayed to the pilot by the fire control computer. This normally is a value, balanced by the available energy of the missile, the seeker's acquisition capabilities, and the maximum powered lifetime of the missile subsystems. The minimum launch range is also a soft termination condition related to fire control, but is determined more by launch safety considerations than anything else.

A combination of many of these parameters will determine the Launch Acceptability Region for a given AAM. In the analysis of friendly systems it is important to apply termination conditions correctly since overstating or understating a weapon's capability does a disservice to the pilot who needs to take the weapon into combat. In the analysis of threat weapons, the same argument holds, but we must also ensure a fair comparison is made between competing weapons. An apples-to-apples comparison is often required to fairly evaluate how threat weapons stack up to our own.

It is critical to understand the application of termination conditions when evaluating LARs. And more importantly, to accurately represent termination conditions within a simulation ensures a balanced comparison is made between competing missiles. In a more philosophical sense, during the analysis of threat weapons, the mastery of termination condition logic allows greater insight into the threat pilot's fire control computer, and although you may have nailed the performance, it is how the threat operator will be able to employ that weapon as limited by his fire control computer which is of greater importance. Whether an AAM has a capability in an air-to-air engagement or not only really matters if the pilot receives a launch queue from his fire control.

LAUNCH ACCEPTABILITY REGIONS

The Launch Acceptability Region, or LAR, is probably the most widely recognized measure of performance associated with AAMs. LARs are known by several other names including Weapon's Employment Zones (WEZ), Launch Zones, and Launch Envelopes. LARs come in various forms, with the common purpose of representing the boundaries and zones of effectiveness for an AAM in a certain scenario. The most common types of boundaries displayed in a LAR are maximum kinematic, minimum kinematic, seeker detection ranges, F-Pole, A-Pole, and E-Pole. LARs can be generated in either the horizontal or vertical plane with either the target or shooter aircraft located at the envelope center. By far, LAR generation is the most frequent use of TRAP by NAIC and our customers, and several utilities have been built into the program to aid the analyst in creating LARs. When applied correctly, LARs can give great insight into a weapon's operational capabilities. [6]

LAR generation is basically a problem reduced to finding the boundary between an area that satisfies a required condition and an area that does not. The reason it has been couched in such generic terms is because of the large variety in the types of LARs a user may be interested in. Below are examples of many common types of kinematic LARs. The boundaries, as discussed in the previous section, are extremely dependent upon the user specified termination conditions. Table 1 lists the termination conditions used for Missile X throughout this paper.

Table 1: Missile X Termination Conditions

Condition	*Value*
Max Guided Flight Time	60 sec
Minimum Flight Time	2.0 sec
Max Seeker Gimbal Angle	60 deg
Max Seeker Gimbal Rate	80 deg/s
Low Missile Mach	Mach 0.9
Low Closing Velocity	150 m/s
Minimum *G*s	1.0 *G*
Maximum Launch Range	40 km
Minimum Launch Range	100 m

In generating LARs with TRAP, there are several methods available to find the boundary. The brute force approach divides the launch space into a grid

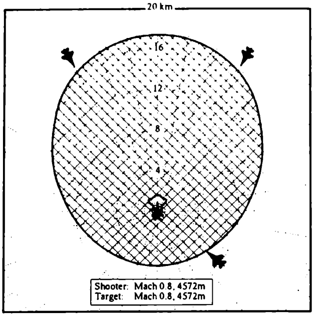


Figure 6: Missile X Target Centered Horizontal LAR

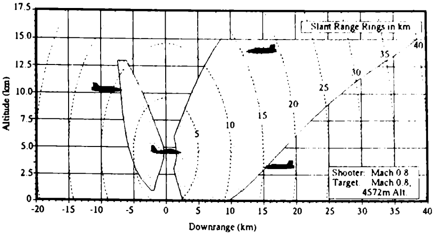


Figure 7: Missile X Target Centered Vertical LAR

and fires a shot at each point within that grid, reporting the result for analysis. The preferred and quicker approach is to apply an iterative scheme using logic based algorithms to search along each aspect to find the minimum/maximum boundaries. Once the LAR boundary point is found, the aspect angle is incremented and the search begins for the next point. This iterative process is repeated for both inner and outer boundaries bounding or defining the LAR.

Horizontal Target Centered LAR:

Figure 6 shows a common example of a target centered LAR for Missile X in the horizontal plane at 15,000 ft, Mach 0.8 with the target and shooter co-altitude and co-speed. The target is located at the center of the polar grid and in this case is non-maneuvering, with the shooter aircraft aimed directly at the target for all aspects. Both minimum and maximum range boundaries are displayed, and the area bounded by each, is shaded, which constitutes the acceptable region within which the missile could be launched and kinematically have the required energy to reach the target, subjected to the termination conditions of Table 1.

Vertical Target Centered LAR:

Figure 7 shows a target centered LAR for Missile X in the vertical plane. In this example, the target remains at a constant altitude and heading for all shots while the shooter's altitude is varied from a user prescribed minimum to a maximum. The maximum and minimum kinematic boundaries are then found for head-on and tail-on engagements respectively. In this particular vertical LAR, the shooter is aimed in aspect, but remains level in elevation so that both aircraft are in level flight.

These first two examples provide the basic boundaries for simple kinematic ranges of the missile. However, using these cases alone, only tell a portion of the story. In both Figure 6 and Figure 7, the target conditions were fixed and in Figure 6, the missile was directly boresighted on the target at all aspects. To gain additional knowledge about an AAM's capabilities, the shooter and target perspectives are now reversed, placing the shooter at the LAR center. By performing both shooter and target centered LARs a more complete picture of a weapon's utility is obtained.

Horizontal Shooter Centered LAR:

Shooter centered LARs help characterize the off-boresight, and lead/lag capabilities of an AAM, by reversing the perspective and placing the shooter at the LAR center. This allows the missile to be evaluated in a more stressful and realistic environment, since having the missile directly boresighted on the target is rare in aerial combat.

To generate a horizontal shooter centered LAR, the launch aircraft is positioned at the simulation origin and the target is moved in an iterative fashion throughout the region in front of the launcher in downrange and crossrange. ***Figure 8*** shows a horizontal shooter centered LAR for Missile X at 15,000ft Mach 0.8 co-altitude, co-speed with both the maximum and minimum range boundaries. For this particular shot, the target aircraft is at a 180 degree heading difference to the shooter. A family of shooter centered horizontal LARs with the target at various headings and maneuvers,

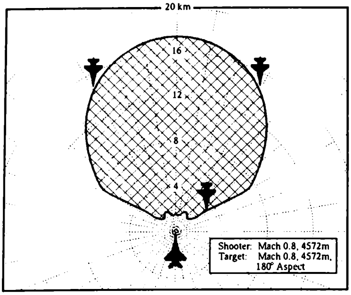


Figure 8: Missile X Shooter Centered Horizontal LAR

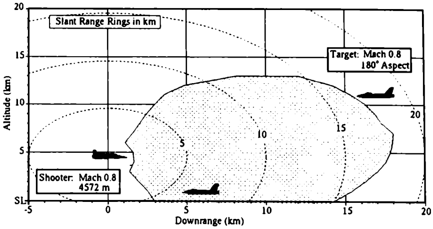


Figure 9: Missile X Shooter Centered Vertical LAR

demonstrates how an AAM's usage can be interpreted from the launch aircraft pilot's perspective.

Vertical Shooter Centered LAR:

To generate a vertical shooter centered LAR, the launch aircraft is positioned at the simulation origin and the target is moved in an iterative fashion throughout the region in front of the launcher in altitude and downrange. ***Figure 9*** shows a vertical shooter centered LAR for Missile X at 15,000ft Mach 0.8 co-altitude, co-speed with both the maximum and minimum range boundaries. For this particular shot, the target aircraft is at a 180 degree heading difference to the shooter.

Since in Figures 8 & 9 the shooter is at the same launch condition, we could interlace the 2-D horizontal and vertical shooter centered LARs to create an approximate 3-D shooter centered Launch Acceptability Volume (LAV?) for Missile X. Although difficult to represent graphically, this in reality is the kinematic capability of the missile in three space. Only for the convenience of pictorial representation have we limited ourselves to the horizontal and vertical planes. A similar argument also applies to Figures 6 & 7 to approximate the 3-D capability for target centered LARs.

Shooter Centered No-Escape LAR:

As we had stated previously, a family of shooter centered horizontal LARs with the target at various headings and maneuvers, demonstrates how an AAM's usage can be interpreted from the launch aircraft pilot's perspective. By creating a series of LARs in this manner we can then overlay each of them, creating a new LAR which is the union of all the individual plots. In effect, this new LAR will be a shooter centered "No-Escape Zone" in the horizontal plane.

Figure 10 shows the process for creating a shooter centered no-escape zone for Missile X. Not every possibility has been presented due to limitations in space, but the major contributors are shown. This process allows us to develop a very powerful measure of effectiveness for an AAM from a pilot's perspective.

Although no comparison system is represented, this volume can expand or shrink greatly depending on the capabilities of the AAM. For older generation dogfight missile, this may actually be a null space since their seekers may limit them to rear hemisphere engagements only. This no-escape zone in effect tells the analyst that anything within the region, regardless of heading or maneuver, is kinematically within a weapon's capability. If we had performed the same analysis in the vertical plane, the combined shooter centered no escape regions could be combined to approximate the 3-D kinematic no-escape volume.

F-POLE, A-POLE, AND E-POLE ANALYSIS

F-Pole, A-Pole, and E-Pole analysis are special types of analysis that are regularly performed with TRAP. Each are important for the tactical analysis of AAMs in engagements. Basic explanations of each are given below.

F-Pole: *F-Pole* is the distance between the shooter aircraft and the target aircraft when the shooter's missile impacts the target. This parameter is of primary interest when evaluating semi-active, radio-frequency (RF) guided missiles. Semi-active RF missiles require the launch aircraft's airborne intercept (AI) radar to continuously illuminate the target in order for the missile to successfully track the target through the entire

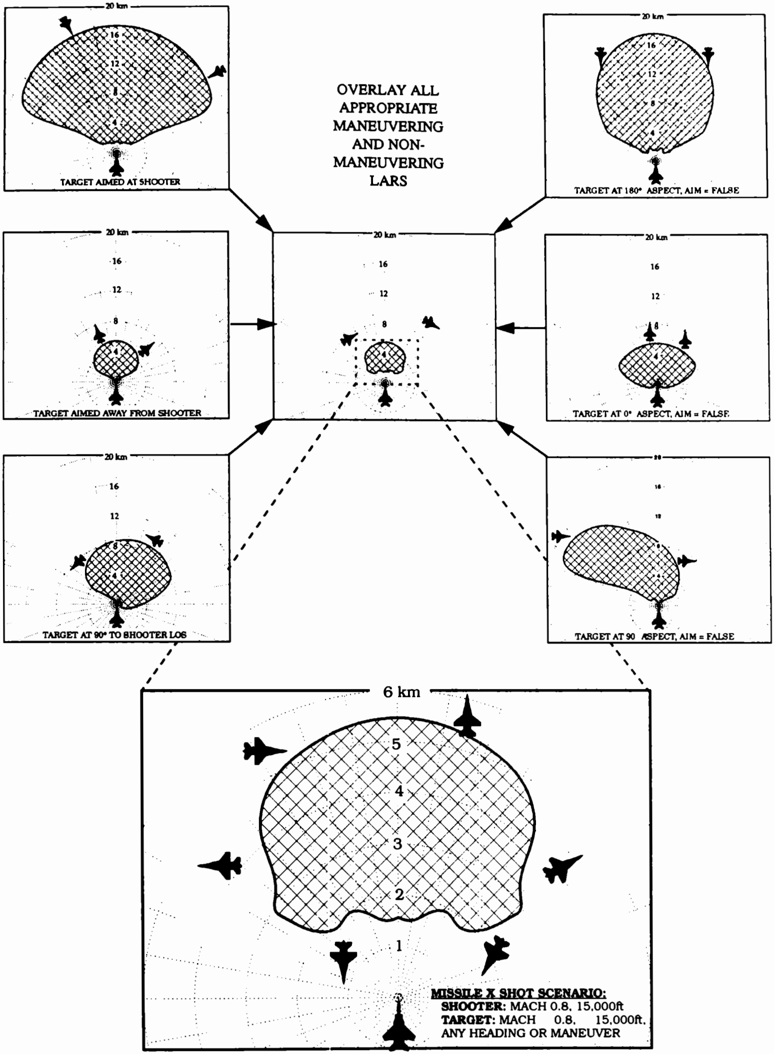

20 km
16
12
8
4
TARGET AIMED AT SHOOTER
OVERLAY ALL
APPROPRIATE
MANEUVERING
AND NON-
MANEUVERING
LARS
20 km
16
12
8
4
TARGET AT 180° ASPECT, AIM = FALSE
20 km
16
12
8
4
TARGET AIMED AWAY FROM SHOOTER
20 km
16
12
8
4
20 km
16
12
8
4
TARGET AT 0° ASPECT, AIM = FALSE
20 km
16
12
8
4
TARGET AT 90° TO SHOOTER LOS
20
16
12
8
4
TARGET AT 90 ASPECT, AIM = FALSE
6 km
5
4
3
2
1
MISSILE X SHOT SCENARIO:
SHOOTER: MACH 0.8, 15,000ft
TARGET: MACH 0.8, 15,000ft,
ANY HEADING OR MANEUVER

Figure 10: Missile X Shooter Centered Horizontal No-Escape LAR

missile flight. This means that once the missile is launched, the shooter is unable to leave the fight until his missile strikes the target aircraft.

To demonstrate a simple F-Pole analysis, consider a head-on 1-v-1 engagement with two aircraft launching semi-active, RF AAMs at the same time towards each other. If we assume both aircraft maintain their speed and heading, the missile with the higher average speed will reach the opponent first, destroying the AI radar and rendering the opponent missile's RF seeker blind. The missile with the higher average velocity in this case translated into an F-Pole advantage.

Figure 3b shows a constant altitude flyout comparison for Missile X vs. Missile Y. Although Missile X has a higher peak velocity, Missile Y has a higher average delta velocity for the entire flyout. Based on this information we could confidently state that for a head-on scenario, under these conditions, Missile Y would always have F-Pole advantage over Missile X, regardless of launch range.

The F-Pole of a missile can also be augmented or maximized by allowing the shooter aircraft to maneuver in such a way as to decrease its rate of closure with the opponent aircraft. Maneuvers such as an offset allow the shooter aircraft to continue to illuminate the opponent with his AI radar, but reduce the rate of closure, thus increasing F-Pole.

- A-Pole: A-Pole is related closely to F-Pole but applies to active radar missiles rather than semi-active. *A-Pole* is defined as the separation between the shooter aircraft and the target aircraft at the time an active radar missile seeker can begin to autonomously track the target. At this point, the shooter aircraft, which likely was providing either illumination or update information concerning the target up until that point, can disengage. The missile will then continue to guide independently using its own radar transmitter to illuminate the target. The tactical advantages of an active radar AAM over a semi-active radar AAM are obvious, since it is much easier for an aircraft shooting an active radar AAM to stay out of harm's way, keeping his distance from the threat target aircraft.

A-Pole is influenced by aircraft maneuvers and the missile's average delta velocity in the same manner that these factors influenced F-Pole. Additionally, the target's signature also contributes to A-Pole. The larger the target's Radar Cross Section (RCS), the sooner the active radar missile can guide autonomously and the shooter can disengage.

E-Pole: or Escape Pole, is the maximum launch range of a missile against a target performing an "escape" maneuver. Escape meaning at missile launch the target attempts to disengage as quickly as possible using the optimum escape maneuver. Normally this E-Pole maneuver would either be a descending or level drag combined with a target maximum acceleration to increase the separation between oncoming missile and target as quickly as possible.

CONCLUSIONS

This paper has demonstrated how a versatile flyout model such as the NAIC developed TRajectory Analysis Program (TRAP), can be exercised effectively to characterize the kinematic performance of Air-to-Air Missiles. This was done by categorizing and presenting the various types of analysis important to the understanding of AAM kinematics. Additionally, the importance of correctly applying missile simulation termination conditions was discussed with a brief description of each. Several examples of the uses of LARs were also presented to demonstrate the most common measures of missile kinematic performance.

However, an underlying premise to the entire paper is a basic assumption concerning the importance of the analyst. Regardless of technological advances, results of modeling and simulation will always require an analyst's scrutiny, understanding that all models inherently have made assumptions and are therefore limited in what they can do for you. Models such as TRAP, are simply tools to help an analyst more fully understand and depict reality, they are not reality themselves. Appreciating this sometimes subtle difference will enable AAM analysts to gain full benefit from the capabilities of modeling and simulation.

REFERENCES

1. Herrmann, Joseph W., "The Trajectory Analysis Program (V3.1a) Synopsis", NAIC/TANW,1995.
2. Borer, A.R., "MAXGEE Program", SDAEW-EM-89-31, 1989.
3 Wolfe Matthew P. , "AVENGER User Manual for Aircraft Performance Analysis", NAIC/TANN, technical paper,1995.
4. Borer, A.R., "Constant Altitude Flyout Plotting Program", SDAEW-EM-88-26, 1988.
5. Mokas, W.H., "Constant *G* Flyouts",NAIC/SCDE technical application paper, 1991.
6. Byram, T.R., Lewis, D.G., Perry, D.D., "Software User's Manual for TRAP 3.1", Battelle, Contract No. F33657-88-D-0076, 1993.

AIAA-96-3490-CP

REUSE OF A J-MASS COMPLIANT MISSILE MODEL AS A SAM MODEL

Larry Lewis
WL/MNSH
Eglin AFB. FL

William J. Bezdek *
McDonnell Douglas Corporation
St. Louis. MO

ABSTRACT

The Joint Modeling and Simulation System (J-MASS) provides an object oriented software development environment that supports the development and reuse of models. configuring the models and the scenario using graphical tools. executing the simulation and post-processing the results. Wright Laboratory's Armament Directorate (WL/MNSH branch) at Eglin AFB has used a beta J-MASS system extensively to develop working three degree-of-freedom (DOF) and six DOF models of generic air-to-surface and air-to-air munitions to support technology development efforts. The munitions models were divided into objects similar to a real world missile including the umbilical, warhead. motor. fuse. seeker. kinematics and aerodynamics. Using J-MASS in the development process was both faster and less costly than tradition methodologies. The original three DOF Eglin air-to-air missile was converted to a surface-to-air (SAM) model at MDA by adding a booster object and command guidance. Since the original model was developed using the J-MASS system. most of the changes needed were in the attributes only. This model is unclassified for demonstration purposes. WL/MNSH has also developed six DOF munitions models based on similar model components. The models are stored in the user's modeling library or the J-MASS Model Simulation Reuse Library (MSRL). The MSRL will contain a variety of verified/validated models (or references to classified models) and model components available to J-MASS users.

BACKGROUND

There have been many missile models developed in the past for use in simulation. Many of these models have been used in either digital or manned simulations at Eglin and McDonnell Douglas Aerospace (MDA). They range from models available in TAC Brawler, ESAMS. and the Trajectory Analysis Program (TRAP) for digital analysis. to real time five degree of freedom (DOF) and six DOF models used in manned simulation.[1 2 3] In particular. MDA has developed a FORTRAN five DOF model which has been used extensively for Sparrow, Sidewinder. AMRAAM, threat missiles, HARMs and Maverick studies. As far as a FORTRAN model can, it has been designed to be reusable and. with new data, can be configured to be used as part of manned simulation studies. This model relies on the user supplying the following input data:

- size. weight. area
- max. g's. trim alpha. aerodynamic coefficients
- control deflections. autopilot gains. natural frequency. and damping
- guidance functions including time-to-guide
- boost / sustain thrust and duration
- thrust vectoring capability
- impulse
- seeker field-of-view. gimbal limits. and off-boresight capability
- countermeasure susceptibility
- time-to-arm. fusing. and lethality
- maximum time-of-flight. min/max velocity

The model has reusable components for the seeker and autopilot which are adapted for each desired configuration. The software engineer uses a copy of the baseline missile model and adapts it for each new configuration. The assumptions made in the model are that:

- it has access to exact information about the target
- any failure modes are determined outside the missile using a random draw
- the point of closest approach between the missile and the target is sufficient to determine the probability of kill (Pk).

* Senior Member. AIAA

This type of model has been used extensively as part of MDA Flight Simulation studies because it operates at a fairly low update frequency of ten hertz and has sufficient fidelity to be used for current and advanced missile studies. The use of this type of model for surface-to-air missiles (SAM)s has not worked out as well because an update rate of at least 40 hertz is normally needed for stability.

Recently, there have been requests to use TRAP models as part of a real time simulation. These models are more CPU intensive, which can affect the update rate of the overall simulation. The advantages to using a common model are that the customers can do their digital analysis using TRAP, the amount of recoding for the real time simulation would be reduced, and revalidation steps could be reduced.

These models provide valuable insight into the primary dependent variables of interest in a missile model which can be used in the development of real time simulation models. The concepts used in traditional software modeling can easily be applied to object oriented models using object oriented techniques like the Joint Modeling and Simulation System (J-MASS).

At WL/MNSH, there have also been many separate stand-alone munitions simulations developed over the years to support technology development in both air-to-air and air-to-surface arenas. Most of these models were developed in FORTRAN and are either specialized one-of-a kind models or based on more common architectures, such as TRAP or similar in-house architectures. They have been used to conduct studies involving the GBU-class of glide bomb munitions, the AGM-130, and Advanced Medium Range Air-to-Air Missile (AMRAAM), as well as laboratory conceived conceptual designs. The disadvantage of all these models are that they are difficult to modify for evaluating new technology, they generally have different and specialized user interfaces (most are non-graphical), and they are not very reusable in different simulation environments.

JMASS 3.0

J-MASS provides an object oriented software development system that supports a library of reusable models and model components as shown on Figure 1 .

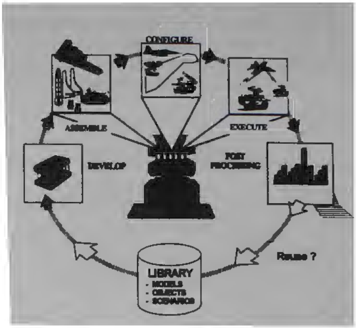


Figure 1 J-MASS System

This system contains a develop mode to graphically create models and model components to be used in the complete simulation. J-MASS also supports configuring the models using external attribute data at run time, so a generic model of a real world system can be compiled and linked into a simulation and then updated with specific data to generate the desired model configuration. The model developer can:

- develop generic model using the graphical J-MASS development tools like the Model Component Development Tool (MCDT) - Figure 2
- reserve attributes for the model that will be set a runtime using external data files
- use the J-MASS automated code generator to create the source code in Ada or C++
- add behavior code to define the generic algorithms
- compile the model and model components
- build the model
- combine the models into teams for execution
- configure the data for the specific version of the model desired - Figure 3
- set-up the models in a simulation using the graphical scenario tools - Figure 4
- execute the simulation while viewing the Combat Battle Monitor - Figure 5
- journal the data
- post-processes the results - Figure 6.

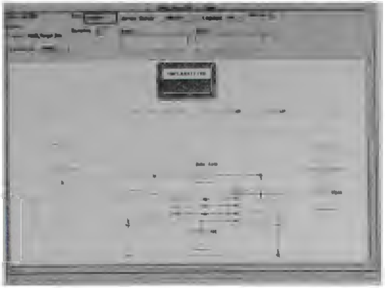
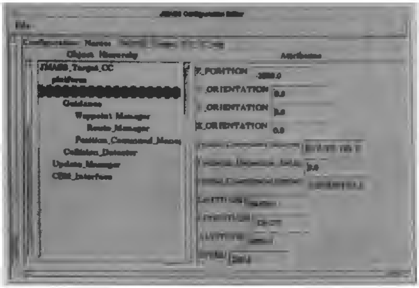


Figure 2 Generic JMASS 3.0 Missile Model



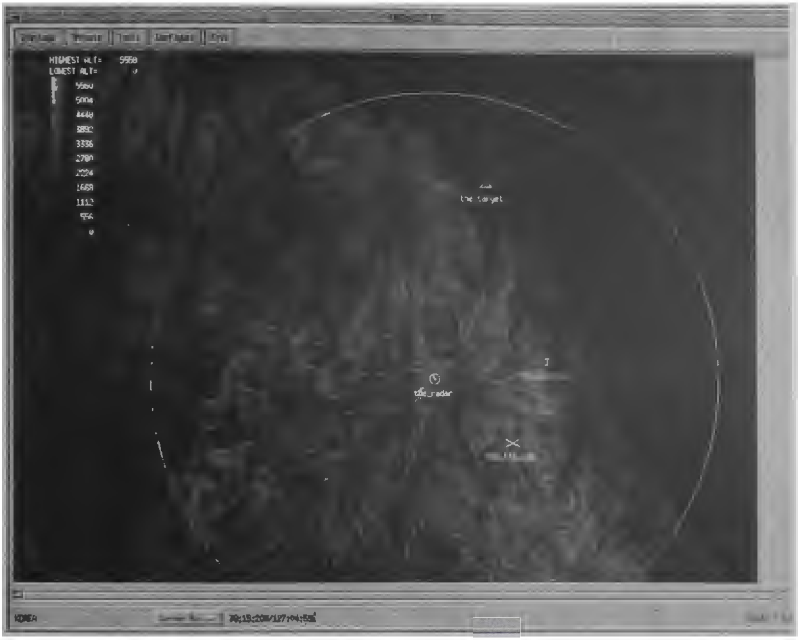
Object Hierarchy

JMASS_Target_CC
platform
[redacted]
Guidance
Waypoint_Manager
Route_Manager
Position_Command_Mana
Collision_Detector
Update_Manager
CBS_Interface

Attributes

Z_POSITION -2000.0
_ORIENTATION 0.0
_ORIENTATION 0.0
X_ORIENTATION 0.0

Figure 3 Configure Attribute Data



HIGHEST ALT= 5560
LOWEST ALT= 0

Figure 4 Configure Scenario

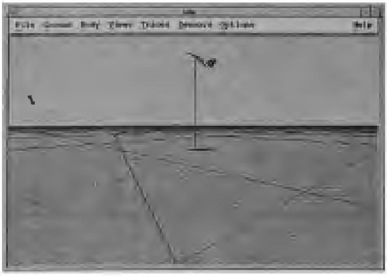


Figure 5 CBM with Missile In-flight

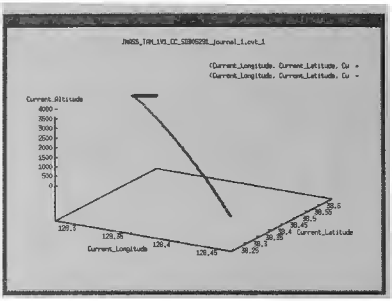


Figure 6 Post-process GNU Plot

Using the graphical tools and modeling and simulation capabilities available with J-MASS, a simulation can quickly and easily be set-up and run similar to the 1 vs. 1 simulation taken from the J-MASS User Manual.[4]

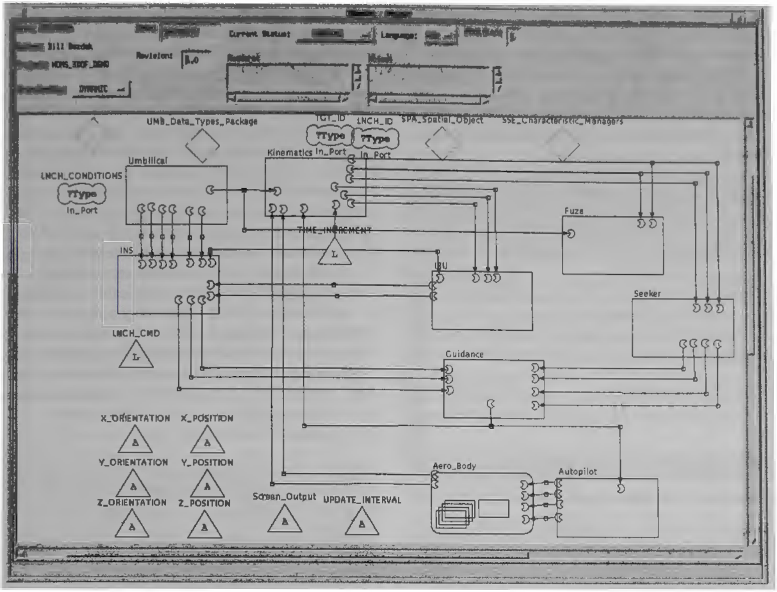
MOMS MISSILE MODEL

WL/MNSH has used a beta version of the J-MASS system extensively to develop working models of generic air-to-air and air-to-surface munitions, beginning with a three degrees-of-freedom (DOF) missile model for experimentation and demonstration purposes. Recently, work continuing with six DOF bomb and missile models which are being used to support technology development for the Small Smart Bomb and for the Dual Range Air-to-Air Missile integrating concepts.

The generic three DOF missile model was divided into "real world" objects including the:

- umbilical
- inertial navigation system
- autopilot
- guidance
- iru
- fuze
- seeker
- aerodynamics

as shown on Figure 7. The motor is currently included in with the guidance. The kinematic object is used to provide the equations-of-motion. WL/MNSH is developing a MOdular Munition Subsystem (MOMS) taxonomy of reusable munitions subsystem models using J-MASS. This extendible library of models will feature standardized subsystem data interfaces for enhanced interchangeability.



Current Status:

Language:

Bill Bezdek

MOMS_3DOF_DEMO

Revision: 1.0

DYNAMIC

Figure 7 MOMS Missile MCDT

The original three DOF model is sometimes called the MOMS three DOF missile model, though it was really a predecessor of what is now becoming the MOMS taxonomy. This three DOF model was constructed primarily for experimenting with J-MASS and for demonstrations. It includes a kinematics section which is set-up for a three degree-of-freedom (DOF) missile, but has been easily updated for a six DOF munitions model. A generic set of algorithmic (behavior) code and the interfaces from object to object were developed, external inputs and outputs defined, a database of parameters to determine the actual operation of the missile were determined, and the model flown. This development process was both faster and less costly than tradition methodologies. The model shown in Figure 7 is the output of J-MASS after the MOMS model was converted from JMASS 2.0d to JMASS 3.0. J-MASS assists the model developer with updating the model as part of the import and translation process. Some manual conversion is required as shown for the port types, like the LAUNCH_CONDITIONS.

MOMS CONVERSION

A surface-to-air missile (SAM) model uses many of the same components as an air-to-air missile model. The original three DOF Eglin air-to-air MOMS missile model was converted to a SAM model by adding a booster object and command guidance. Since the original model was developed in the J-MASS architecture, most of the changes were in the data only. The resulting model is unclassified for demonstration purposes, provides an example of a "good" software development processes in J-MASS, and provides a simple model that can be used in the early stages of development for similar armament.

The three DOF SAM model uses the MOMS three DOF data set supplemented with data from unclassified sources to simulate an SA-6, SA-8, SA-10 or SA-12. The final MCDT diagram for the model is similar to the MOMS model on Figure 7.

The guidance model takes the range data to the target along with range rate, relative geometry of the target relative to the missile including line-of-sight rate and generates the acceleration commands that are output to the autopilot and kinematics based on calculations from the seeker and the inertial navigation system. The guidance MCDT diagram for the missile is shown on Figure 8.

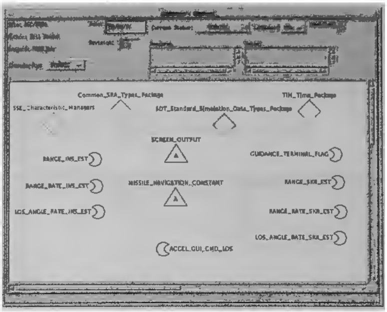


Figure 8 MOMS Missile Guidance MCDT

The aerodynamics shows an example of the top-down hierarchy of J-MASS. The aerodynamic object is an assembly in J-MASS and contains additional details at the next lower level. By using the mouse on Figure 7 to "view the details", a view of the Aero_Body object assembly is displayed as shown on Figure 9. It contains lower level objects for each fin actuator and the aerodynamics. Each fin actuator is an instance of the same actuator model.

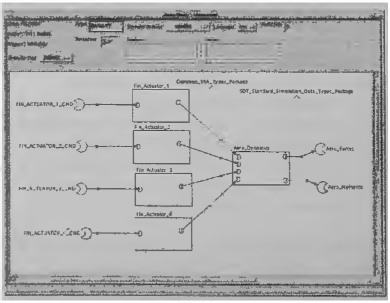


Figure 9 MOMS Aerodynamic MCDT

SAM THREE DOF TESTING

The testing of the three DOF SAM model involves replacing the existing missile in the J-MASS tutorials with the SAM3DOF. The missile receives launch commands and command guidance from the J-MASS C4I Van as part of a one-on-one missile test which is shown in the J-MASS User Manual. The flight path and location of the models are the same ones used in the tutorials. The model configuration data also needs to be updated for the missile which was built as an air-to-air missile to function as a surface-to-air missile. The configuration data screen is shown on Figure 10. The C4I Van should be enhanced to provide decision logic for launch besides range. Potential versions of a decision package that might be applicable is described in another paper. [5] Even though only a single instance of the missile was launched, there is no reason why several instances of the missile could not be launched. J-MASS supports multiple instances of a model. The logic for the C4I Van would need to be updated. Using J-MASS for multiple targets is also described in another paper. [6]

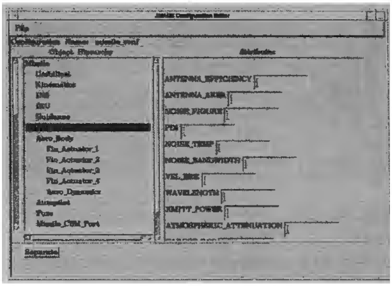


Figure 10 MOMS Configuration Data

SIX DOF BOMB MODEL

Since the initial 3 DOF missile model was developed, WL/MNSH developed a Guided Bomb Player (GBP), which is a generic six DOF model of a basic non-powered guided bomb. It was designed to have reasonably high fidelity aerodynamics and a modular design which would allow easy substitution of subsystem components. The GBP was modified in a very short period of time to represent a conceptual Small Smart Bomb (SSB) and has been successfully used for the past year to support the Miniaturized Munition Technology (MMT) program at WL/MNSH. In addition, another modified version of the GBP will be developed to represent a JDAM to help guide laboratory technology developments for that system. Most importantly, the GBP added critical reusable components to the MOMS J-MASS library and is serving as the foundation for all subsequent six DOF munitions modeling at WL/MNSH. These baseline six DOF models are currently being upgraded to J-MASS 3.0. No MCDT diagrams were available at the time of this report.

SIX DOF MISSILE MODEL

By adding a rocket motor to the basic GBP and making some adjustments to the aerodynamics coefficients and other subsystem attributes, WL/MNSH created the foundation for a basic six DOF air-to-air missile model in a very short period of time. Final components are being completed at this time to represent a generic air-to-air missile player (AAMP) which will be used to support the Dual Range Air-to-Air Missile integrating concept studies.

Classified models of real world systems are being developed for use with J-MASS. These models have classified data and in some cases classified algorithms. A classified AMRAAM missile model, using the AAMP, is also being completed this summer to allow laboratory technology trade-off studies to be conducted for that system. A methodology has been developed to store J-MASS models in a Model Simulation Reuse Library (MSRL), which contains a variety of verified/validated models (or references to classified models) and model components available to J-MASS users or in the user's local library.

REUSABILITY OF SIX DOF

Almost 90% of the original GBP components were reused to build the SSB and AAMP munitions models in a very short time. The key to this reusability is careful design of subsystem interfaces (an important idea in the development of MOMS components) so that any subsystem model with the same parameter I/O can be used in place of an existing model (within limits of reasonable engineering design). This not only allows experimentation with different conceptual designs, but also allows building more than one model of the same subsystem for purposes of having different model fidelities easily available. Also, by carefully designing the core dynamics and aerodynamics models, these can be almost 100% shared among all six DOF bomb and missile models. It is expected that using MOMS compatible J-MASS models over the next several years for munitions system and subsystem development will save several million dollars over traditional modeling approaches.

CONCLUSION

J-MASS provides an object oriented methodology for model development that supports reuse. A simple missile model was used as a baseline which was linked to the simulation through a C4I Van which received target locations via a radar linked to the Van. The target track was used to supply a launch command and command guidance to the missile which intercepts the target.

Using the MOMS missile model provided a much more complete model to use for testing that provides better, more meaningful results. The basic MOMS missile model is a three degree-of-freedom (DOF) model which was originally set-up as an air-to-air missile. With only minor modifications, the MOMS model was upgraded to operate in JMASS 3.0, converted to a surface-to-air (SAM) missile model, and flown in the same scenarios developed for the simpler model.

Missile and bomb models have also been built using J-MASS that provide complete six DOF dynamics for higher fidelity testing. In addition, the six DOF models can be used to simulate munitions like the AMRAAM and JDAM as well as new munitions concepts such as the Small Smart Bomb and Dual Range Air-to-Air Missile. J-MASS modeling provides a cost effective methodology for software development and will effectively reduce the cost of conceptual weapon development by enabling rapid model prototyping of concepts and time efficient trade studies.

FUTURE WORK

The use of J-MASS and object oriented design techniques for air-to-air missile models have already produced models which contain reusable components which are applicable for a variety of similar applications like surface-to-air and air-to-surface weapons. Using an object oriented breakdown of the real world system into J-MASS objects, the resulting models and model components have been extended into other areas like glide bombs and similar applications. In many cases, this can be done by modifying the attributes and not requiring recompilation of the generic model. One area where the use of a Field Extensible tools would be helpful is with the autopilot and control system logic of the model. One tool that is gaining wide acceptance at MDA is Matrix X from ISI. Matrix X has been applied to several missile and aircraft models with good results. [7] This class of tools contains graphical screens which allow the engineer to design the algorithms on the screen, perform analysis, and verify the operation of the algorithms in a simulation. An example is shown on Figure 11. Notice that the object breakdown for Matrix X is very similar to the J-MASS object oriented breakdown. It would be useful if the capabilities of J-MASS to provide the graphical tools to break down a real world model into components, to configure the models into a scenario with other models, execute and post-process the results could be combined with the graphical algorithm development of Matrix X. In both cases, the models contain automatic code generators for Ada. J-MASS also supports C++ and Matrix X supports "C".

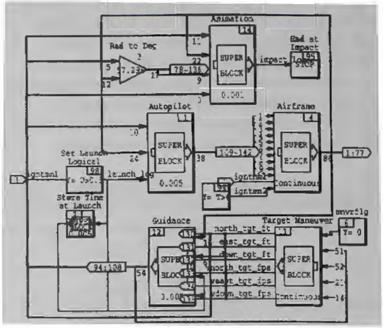


Figure 11 Matrix X Diagram

The second area that should be examined is rehosting J-MASS from workstations to PCs. J-MASS was designed to be POSIX compliant which allows it to be used on several UNIX workstations. A combination of workstation servers and Xterms is fairly cost effective, but still requires the server CPU to execute the simulation which can become a bottleneck. The PCs can be configured with software like eXceed for Windows by Hummingbird Communication, Ltd. to use the PC as an Xterm as well.[8] The current processors available in PCs are rivaling the speed of some workstations. J-MASS already has the capability, using Unified Message Passing, to execute on a network of heterogeneous processors using either Ada or C++ as well as using batch processing capabilities to distribute the runs over a variety of CPUs.[9] If the J-MASS models could be executed on the PCs, all the CPUs available on the network could be applied to the J-MASS simulation.

REFERENCES

[1] TAC Brawler Users Manual, 1993.

[2] ESAMS User Manual, 1994.

[3] Hermann, J.W., "Air-to-Air Missile Engagement Analysis using the TRajectory Analysis Program (TRAP)", AIAA 96-3489.

[4] J-MASS User Manual, Vol. II, "Getting Started", J-MASS 3.0a, 15 April 1996.

[5] Bezdek, W.B. and Reidelberger, E.A., "Vehicle Interactive Digital Pilot Model Using J-MASS", AIAA 96-3494, July, 1996.

[6] Bezdek, W.B. and Herr, C., "Use of J-MASS as a Many-on-Many Engagement Model", AIAA 96-3492, July, 1996.

[7] Matrix X, ISI, User Manual, 1995.

[8] eXceed for Windows, Hummingbird Communications, Ltd., User Guide, 1995.

[9] Beavin, W., Emrich, T. and Bezdek, W., "J-MASS DIS Interface Using J-MASS", NAECON, May 1995.

AIAA-96-3492-CP

USE OF J-MASS AS A MANY-ON-MANY ENGAGEMENT MODEL

William J. Bezdek [*]
McDonnell Douglas Corporation
St. Louis, MO

Christine S. Herr
McDonnell Douglas Corporation
St. Louis, MO

ABSTRACT

The Joint Modeling and Simulation System (J-MASS) provides an object-oriented modeling and simulation (M&S) system that can be used to configure and execute a many-on-many engagement. J-MASS currently supports few-on-few engineering analysis and simulation with varying model level-of-detail, supports the development of new object-oriented models, the set-up of multiple instances of an object quickly and easily, the configuration of the desired data for each model, the aggregation of model entities together, and graphical placement of the models and aggregates into a scenario anywhere in the world. To turn a group of model entities and aggregates into a useful simulation, the user must not only furnish the models, but the decision logic, command and control, and the direct control of entities in the simulation. This is done in J-MASS, not by having an overall control model, but by distributing the strategy, operation, and control to the individual elements of the simulation, with group command and force structure command modeled similar to the real world. A simulation was set-up and run using J-MASS with multiple simple models interacting in a many-on-many simulation.

BACKGROUND

There are currently simulations that support many-on-many mission and theater level engagements including Extended Air Defense Simulation (EADSIM), Suppressor, and TAC Brawler to name a few. These simulations can contain over 100 model entities and support event-based and in some cases real time simulations.

EADSIM is used by operational commanders, trainers, and analysts to model the performance and predict the effectiveness of a wide range of threats. [1] EADSIM uses two standard interface protocols to confederate, the Aggregate Level Simulation Protocol (ALSP) and the Distributed Interactive Simulation (DIS) protocol. EADSIM is a wargaming tool for training operators, battle staffs, and field commanders.

Suppressor is a mission-level simulation model for many-on-many analysis of the effectiveness and survivability of weapon systems in combat. It uses a flexible method of representing elements of the engagement and their interactions. Players may include surface-to-air missile systems, anti-aircraft artillery (AAA), fighter aircraft, bombers, anti-radiation missiles (HARM), electronic combat systems, and naval vessels. Multi-level C3 networks can be defined to model intercommunications and time delays between elements. [2] Suppressor has been updated and installed at Pax River to support real time simulation in addition to event based non-real time simulation as SWEG.

TAC Brawler is a government owned air-to-air engagement model sponsored by Air Force Studies and Analysis. Known for its modeling of pilot decision logic, TAC Brawler is a Monte Carlo model with high detail representations of avionics, weapons, aircraft performance, and situational awareness. Results from TAC Brawler resemble those of an instrumented test range, and reflect the fact that no two engagements are the same, but that a pattern will emerge over time. [3]

EMERGING MODELING AND SIMULATION ENVIRONMENTS

The DOD is moving toward an integrated modeling and simulation (M&S) system which supports simulations with a common set of models and provides training support for a wide range of users. The threats used in these simulations will be verified and validated by their military sponsors. These initiatives are in four areas:

- J-MASS
- JWARS
- JSIMS

[*] Senior Member

- HLA

J-MASS - The Joint Modeling and Simulation System (J-MASS) is a common computer-based modeling and simulation system that supports activities including research and development, acquisition, analysis, and test and evaluation. The primary focus of J-MASS is test and evaluation. J-MASS supports the hierarchy of applications at the engagement level, with strong support of the engineering and mission levels. When developed, it will be a DOD standard modeling and simulation system compliant with the evolving High Level Architecture (HLA). [4]

J-MASS provides for the development, execution, and post-processing of simulations. It supports the development of digital objects representing friendly systems, threat systems, environments, and cognitive human operator models at varying levels-of-detail. It supports the configuration and execution of simulations using these objects and the post-processing of simulation results. J-MASS defines a minimum set of standards, common protocols, and a well-defined description of how to determine the nature of the information that should be passed between models to form a simulation.

JWARS - The Joint Warfare System (JWARS) will provide the infrastructure to simulate large campaign simulations used for acquisition and planning. JWARS can be used by operational commanders and analysts to model the performance and predict the effectiveness of a wide range of weapon systems and threats. [5]

JSIMS - The Joint Simulation System (JSIMS) will be a single, seamlessly integrated simulation environment. It will include a core infrastructure and mission space objects maintained in a common repository. These can be composed to create a simulation capability to support Joint or Service training, rehearsal, or education objectives.

JSIMS will simulate all forces and activities and will respond to the training audience. JSIMS will be able to compose a simulation of only those forces and activities dictated by the requirements of the exercise or simulate multiple alliances and/or coalitions. JSIMS will be used to simulate activities of national and tactical intelligence, surveillance, and reconnaissance assets including tasking, collection, assessment, and dissemination. [6]

HLA - The high level architecture (HLA) will provide a communications infrastructure to facilitate interoperability between different simulations and the ability to "federate" existing simulations in order to improve reuse. HLA will use simulation applications constructed from modular components with well-defined functionality and interfaces.

HLA will provide interoperability within simulations, among simulations of a federation, or across functional communities. HLA will support a broad range of functional areas including training, contingency planning, analysis and acquisition. Applicable simulations involve software only representations, man-in-the-loop simulations, and live components. [7]

J-MASS MODELING AND SIMULATION

This paper will focus on the use of J-MASS to provide a simulation environment that supports engineering and analysis with a few players using dynamic, multiple level-of-detail models that can be expanded, using basic J-MASS concepts, to address larger many-on-many simulations which provides the base for the other types of simulation in the J-MASS environment. Issues will be discussed, a J-MASS simulation set-up, run, and analyzed; lessons learned; and further work planned.

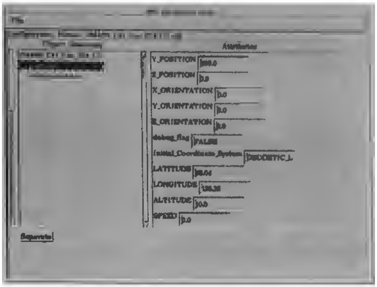
J-MASS is an object-oriented modeling and simulation (M&S) system that supports the development and execution of model entities from complete aircraft, radars and weapon systems (which J-MASS refers to as players) to multi-team distributed simulations that can support a many-on-many engagement. The methodologies J-MASS uses to support a wide range of M&S needs include:

- real world, object-oriented design
- graphical programming environment for Ada or C++ using automatic code generators
- software structural model that supports reuse
- graphical configure capability to support scenario set-up and model aggregation
- event-based or real time processing
- distributed heterogeneous network
- batch processing
- graphical execution monitor
- post-processing analysis and playback
- library of verified/validated model entities and components
- support of Sun and SGI platforms
- set of digital/interactive/manned simulation models/interfaces to support model behavior decision making, and command and control
- user manuals, training, and operational support

JMASS 3.0A

J-MASS currently supports few-on-few engineering analysis and simulation that provide for a wide range of weapon system analysis and threat development needs. Included with J-MASS 3.0a is a 4V4 engagement tutorial that contains the step-by-step procedure to set-up, run and analyze the simulation.[8] With J-MASS, the analyst is able to:

- view or modify the generic model using the graphical J-MASS development tools (Figure 1) : i.e. a sample target, missile, radar and C4I van
- modify an existing model
- build a complete new model
- assign attributes with default values that can be modified at runtime
- use the J-MASS automated code generator to modify or create new source code
- add behavior code to define the generic algorithms, if needed
- compile the model components
- combine the models into teams for execution
- build the simulation
- configure the data for the specific version of the model desired (Figure 2)
- configure the models into a simulation using graphical scenario tools (Figure 3)
- aggregate the models together and place them into the scenario
- execute the simulation while viewing the Combat Battle Monitor (Figure 4)
- journal the data for further analysis
- post-process the results (Figure 5)



Y_POSITION [illegible]
Z_POSITION 0.0
X_ORIENTATION 0.0
Y_ORIENTATION 0.0
Z_ORIENTATION 0.0
debug_flag FALSE
Initial_Coordinate_System GEODETIC_L
LATITUDE [illegible]
LONGITUDE [illegible]
ALTITUDE 10.0
SPEED 0.0

Figure 2 Configure Attribute Data

Figure 3 Configure Scenario

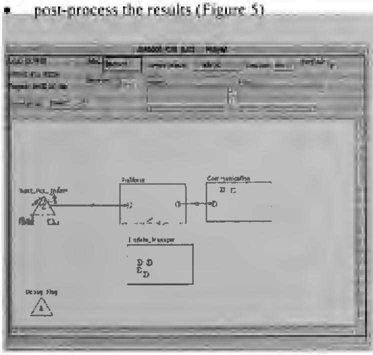


Figure 1 Generic JMASS 3.0a C4I Van

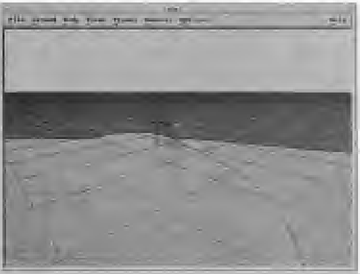


Figure 4 CBM with Missiles In-flight

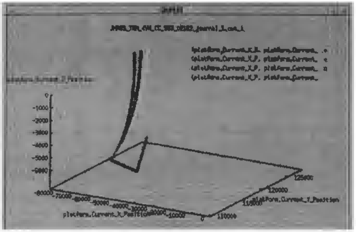


Figure 5 Post-process GNU Plot

Using the combination of graphical tools and the modeling and simulation system available with J-MASS, a simulation can quickly and easily be set-up similar to the 4V4 simulation. The basic components of the tutorial can be modified without having to recompile and rebuild the simulation to add additional instances of the players, aggregate the players, modify the scenario, run the simulation, and analyze the new results.

J-MASS USE IN THE DEVELOPMENT OF A MANY-ON-MANY SIMULATION

J-MASS was used to expand the few-on-few simulation into a many-on many simulation using tools already available in J-MASS. The key concepts include:

- allowing multiple models with different levels-of-detail to be used as a single instance
- having models aggregated into groups of real world entities
- having control (digital pilot) models which allow the user to set-up the behavior of the models and model aggregates
- having tools to graphically configure the scenario for the additional players and aggregated players
- having tools to select and save the desired data for analysis
- having graphical tools to monitor the operation of a many-on-many simulation

Some of the capabilities of J-MASS used to accomplish a many-on-many simulation are discussed below.

J-MASS LIBRARY - Many of the model entities desired may be available from the J-MASS Model Simulation and Reuse Library (MSRL) which will contain verified/validated models available to J-MASS users. Classified models only show the model description and contact information to establish a need-to-know. The MSRL may be accessed by using Internet or modem to contact and browse the library contents. A J-MASS user would download the desired models into a local modeling library (ML). This local ML is very important to J-MASS because it houses the system and user files which make-up the models available to use in a J-MASS simulation as show on Figure 6. The combination of a standards library and a user library allows the user to use existing models and scenarios for a simulation, but at the same time allows the user to:

- have a private copy of the models
- reuse of the existing models and modify the data
- modified models can be developed

so excursions can be tried without affecting the baseline.

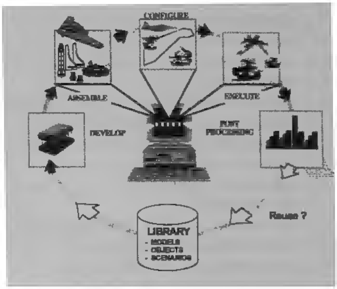


Figure 6 J-MASS System Concept

J-MASS MODEL DEVELOPMENT -The user has a high degree of flexibility in building models with different levels-of-detail, some of which can be selected at configure time. For the many-on-many engagement model, as in other simulations, it is important for all models to have common interfaces, so one model can easily be replaced by another with a different level-of-detail as shown on Figure 7. J-MASS uses ports to allow a model to interface with other models and supports an Application Program Interface (API) to allow modelers to design models which can be integrated into a simulation.[9]

Figure 7 J-MASS Level-of-Detail

DYNAMIC LEVEL-OF-DETAIL CONTROL - Users may want to dynamically switch model entity levels-of-detail during a run. This is possible with J-MASS by compiling and building teams that contain all the levels of detail needed using one of several methodologies. The one investigated involved downgrading the individual players to assemblies and adding a new player broker which interacts with the user/simulation which can dynamically select the level-of-detail desired. This is done using a port to the player broker which receives inputs on the version of the player to use and calls the appropriate assembly. A multi-level target, capable of running in varying levels of fidelity, was developed using J-MASS. This multi-level target player was developed by adapting the sample target player that is delivered as part of the J-MASS system.

Figure 8 and Figure 9 show the top-level MCDT diagrams for the sample target player and the multi-level target player, respectively. The sample target player that is provided with J-MASS is implemented as a player that contains two assemblies – one that implements the control and one that implements the platform. For the multi-level version of this target, the sample target player was demoted to an assembly.

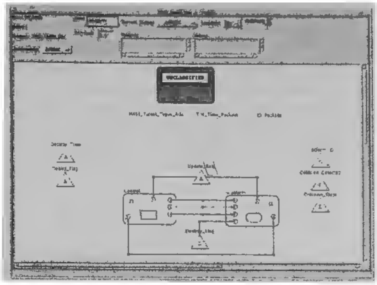


Figure 8 Target Player MCDT Diagram

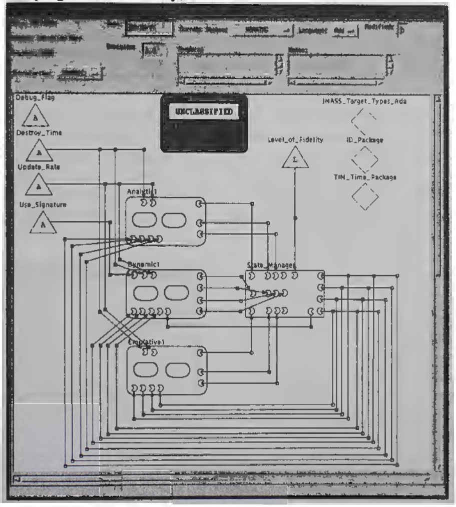


Figure 9 Multi-Level Target MCDT Diagram

The multi-level target was implemented as a player (the top-level J-MASS modeling construct) which contained several assemblies. The player was responsible for initializing the constituent model components and for determining the level of fidelity to be used for each update. Three of the assemblies contained in the multi-level target player were responsible for implementation the algorithms used for each level of fidelity. These assemblies were named Analytic1, Dynamic1, and Emulative1 and were adaptations of the original sample target player. The final assembly, State_Manager, manages the overall state of the target.

Maintaining the integrity of the state of the multi-level target player was a prime consideration when modifying the design of the J-MASS sample target player. A J-MASS model, whether implemented as a player, assembly, element, or component, assumes that it is responsible for maintaining its own state. It assumes that it is the sole implementation code for a particular instance of the model. This is generally a valid assumption. However, in this example, a single instance of the target could be updated with any one of three possible sets of code. It could be updated using the Analytic1 assembly, using the Dynamic1 assembly, or by using the Emulative1 assembly. Because of this, it was not possible to allow the state data to reside in any assemblies and still maintain integrity of the model data as the levels of fidelity were changed. The State_Manager assembly was designed to handle the problem of maintaining the state of the target.

At the beginning of an update, the State_Manager provides the other assemblies with the current state of the target player. At the end of an update, whichever assembly was invoked to update the target player at the appropriate level of fidelity provides the State_Manager with the updated state. One of the duties of the State_Manager is to determine which state data is valid and provide default values for the non-valid state data. This example assumes that the data defining the emulative state of the player is a superset of the data defining the dynamic state of the player which, in turn, is a superset of the data defining the analytic state of the player. If the level of fidelity of the player is downgraded, from emulative to dynamic for example, the State_Manager needs to recognize that the data set by the emulative assembly is no longer being updated and, therefore, is no longer valid. Consequently, if the level of fidelity is ever upgraded back to emulative, the data that is unique to the emulative state needs to be reinitialized back to their default values. A similar situation occurs switching between analytic and dynamic, or analytic and emulative.

In order to work with the State_Manager, the J-MASS sample target player, along with its constituent components, need to be modified to accept previous state data as input parameters rather than maintaining the state internally. This design modification had the advantage that it allowed any of the newly designed assemblies to be used to update a single instance of the target, but it had the following disadvantages: it increased the number of parameters required to update the individual model components and it required the top-most model component to have parameters that defined the state of the inner-most component.

Figure 10 shows the MCDT diagram for the Dynamic1 assembly. This diagram shows that the dynamic assembly contains locals, includes, inputs, outputs, and subassemblies.

- The locals are used for temporary storage. They are not used to define the internal state of the assembly; all state data is passed via parameters.
- The "includes" show the other software components in the J-MASS architecture that are used by this model.
- The inputs contain the data needed for this update (e.g., the update time), and the previous state of the multi-level target player.
- The outputs contain the updated state of the player.
- The dynamic assembly, as well as the analytic and emulative assemblies, is decomposed into two subassemblies. These assemblies implement the control and platform for the target. Each of the top-level assemblies contain a control and platform that provide implementations appropriate for the intended level of fidelity.

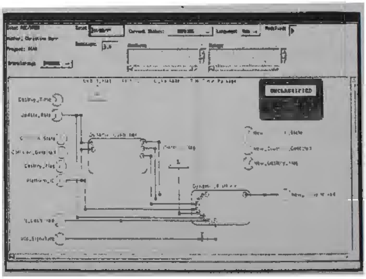


Figure 10 Dynamic1 Assembly

The controller in the dynamic assembly is responsible for reading the digital pilot inputs and converting them into commands understood by the vehicle. The platform in the dynamic assembly is responsible for movement of the vehicle.

Two methods were employed to control the level of fidelity used for a particular update. First, an attribute was placed in the multi-level target player that allowed it to be initialized for a particular level of fidelity. Second, a port was added to the player that allowed it to be sent a message which would cause it to switch to the fidelity level specified in the message. If a message was never received during a simulation run, the multi-level player would continue to execute at the level of fidelity specified by the attribute.

For the purpose of this example, an over-simplified approach was taken in transitioning the multi-level player between the various levels of fidelity. A second player in the simulation was developed that, at specified intervals, sent a message to the multi-level target player telling it the new level to use.

CONTROL OF MODELS IN A J-MASS SIMULATION - The control of the models used in a J-MASS simulation is done using digital pilot models for the individual players.[10] These digital pilots allow the user to set-up the scenario to:

- be completely automated (i.e. follow waypoints as shown previously on Figure 3),
- be interactive with the rest of the simulation models (i.e. an aggressive target),
- have interactive control by the user (i.e. the user controls maneuver selection), or
- allow the user to have complete control of a model or group of models.

The basic concept is that individual models have a set of desired outcomes based on:

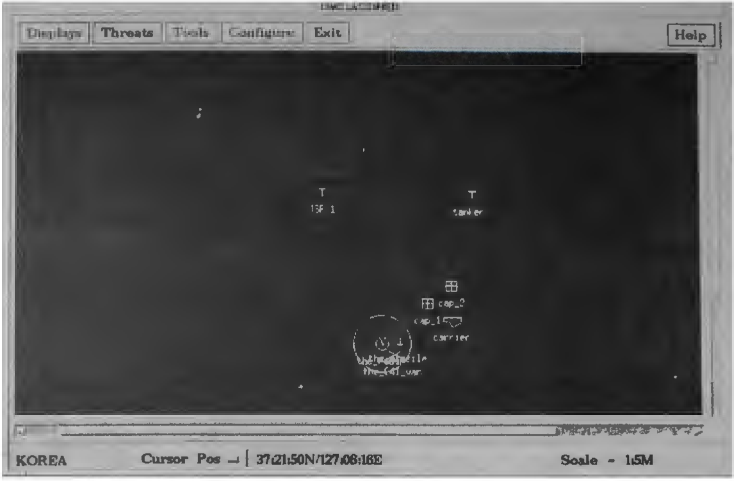
- order of battle
- inputs from a group command player
- expectations
- knowledge of current situation

The actions taken may seem random at first until order is added using the criterion shown above.[11] Command, control, and communications should be made available for all digital, interactive, or manned players. Using this paradigm, a J-MASS simulation could be used in a variety of ways, depending on the method of control chosen.

J-MASS could be used as part of a simulation to train the battle staff by inputting commands from decisions of the actual commanders, providing analysis using digital and interactive commands to see expected outcomes, or providing analysis on the effects of changing several components during repeatable simulations. J-MASS supports digital batch analysis with several experiment manager tools.

J-MASS SCENARIO SET-UP FOR A MANY-ON MANY SIMULATION - One of the unique capabilities of J-MASS is to set-up multiple instances of an object (model entity) quickly and easily, configure the desired data for each of the models, aggregate model entities together, and graphically place the model entities and aggregates into a scenario anywhere in the world.

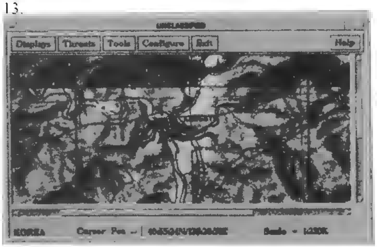
The Scenario Development tool was used to add players and aggregates of players to set-up a larger scenario reusing the same players in deferent roles as shown on Figure 11.



KOREA Cursor Pos → 37:21:50N/127:08:16E Scale - 1:5M

Figure 11 Scenario with Additional Players

As one would expect in a large scenario, it is difficult to view the specific configuration of the models in the scenario. The Scenario Development tool provides the capability to zoom in on part of the scenario and add maps or DTED elevation data as shown on Figure 12 and Figure 13.



KOREA Cursor Pos → [illegible] Scale - 1:250K

Figure 12 Threat and Target Area

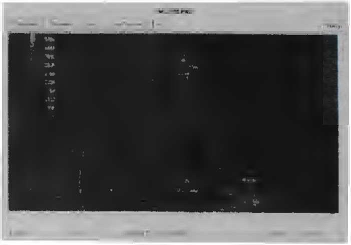


Figure 13 SAM Defenses in S. Korea

As part of the scenario, Figure 11 shows an unclassified model, called JSF, penetrating the northern SAM defenses. That model originated from the carrier off the coast, flew a waypoint to the tanker, went feet dry to attack the target in the heavily defended area to the north. The target and associated waypoint can be viewed on the DTED map as shown on Figure 14.

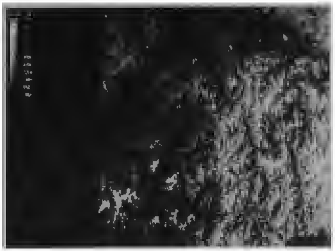


Figure 14 Target Waypoint

J-MASS TESTING Using control input coming from all digital models developed in J-MASS, a series of simulation runs were made. J-MASS Tutorial 5 from the J-MASS User Manual was used for the testing done for the many-on-many simulation with additional players added. The testing used the Scenario Development tool to set-up waypoints for the targets to follow. The targets were used along with the existing C4I Van, which receives target inputs from a sample radar and provides the launch and command guidance for groups of surface-to-air missiles.   The laydown for the southern group of combatants are shown on Figure 15.

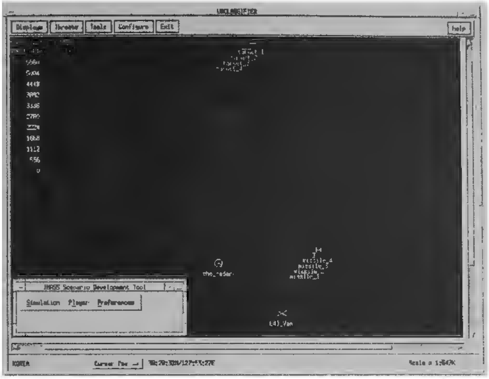


Figure 15 Partial Test Scenario

In this scenario, several groups of missiles are aggregated and placed into the simulation, both on the blue team and the red team. The level-of-detail of the JSF target was varied using the level-of-detail attribute.

The scenario was executed using J-MASS as an event based simulation on a Sun SPARC 20 computer. A snapshot of the Combat Battle Monitor, a J-MASS visualization tool, shows some of the models during the simulation on Figure 16 . The runtime of the simulation could be improved if additional CPUs are available to distribute the team between the available processors. [12]

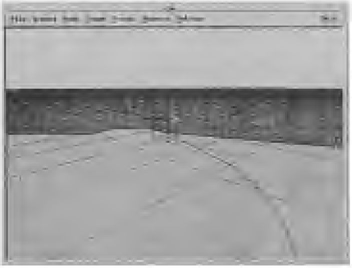


Figure 16 CBM for Multi-Target Scenario

CONCLUSION

The purpose of developing models with multiple levels-of-detail is to provide a range of models which can be mixed and matched during the simulation to reduce cost while retaining the capability to perform digital, interactive, and manned simulation; event-based models mixed with real time models; and event-based simulations using reusable, modular software design. The new capabilities developed for use in J-MASS permit design studies to take advantage of reusable, modular software to perform digital, interactive and manned piloted studies using common software and to allow the software to be converted during the run from an analytic level of fidelity to a detailed model and back again.

There are several advantages using J-MASS as the modeling and simulation environment in the process of developing a many-on-many simulation environment including:

- graphical tools to build object oriented models with common interfaces at different levels-of-detail
- automatic code generators to support Ada and C++ models with the same block diagram
- a library of objects and complete models like aircraft, radars, missiles
- a configure capability so the models can easily be modified by the without recompiling
- a graphical configuration mode to place the models into a scenario anywhere in the world
- graphical execution and post-processing tools to monitor simulation results
- a library of verified / validated models placed in the government reuse library
- customer support, training and user manuals
- models developed in J-MASS that can be exported and used in other simulations

The models used have simplified command and control logic based on the digital pilot designed / developed for J-MASS as discussed in another paper. In general, the models used in this simulation did not have the command and control structure to allow the models to interact in different ways.

Using the M&S techniques of J-MASS, simulation models have been developed at various levels of detail which are usable for low level analysis, as dynamics simulation models, and as detailed models of the real world system. In addition, the design supports digital, interactive and manned simulation at each level-of-detail as part of a simulation using J-MASS.

FUTURE WORK

The use of J-MASS for a many-on-many simulation uses simplified, unclassified test models for the different objects. Realistic models should be substituted for the test models and the simulation rerun.

The command and control logic needs to be added for the simulation to allow digital control of the models used in the simulation using a rule-based decision tree. There are several ways to add that capability. One approach would be to utilize the existing command and control structure of a large simulation program, like Suppressor. Another would be to add additional inputs to each model to allow that model to get command and control logic from the next level of the simulation. This would provide a hierarchy of command and control and allow more and more aggregated and centralized commands at each level of the simulation. Lacking, or losing command and control, the units would continue based on their last set of instructions or their internal operating principles.

REFERENCES

[1] US Army Space and Strategic Defense Command, "EADSIM", 1995.

[2] Suppressor User Manual, 1993.

[3] TAC Brawler User Manual, 1994.

[4] J-MASS ORD, March 1996 (Draft)

[5] JWARS Briefing, Washington D.C. May 1996.

[6] TRD 041896, Joint Simulation System (JSIMS) Joint Program Office (JPO), Technical Requirements Document, 4/19/96 (Draft)

[7] HLA Briefing, Dr. Judith Dahlman, Dec. 1994.

[8] J-MASS-GS-3.0a, J-MASS User Manuals - Volume II - Getting Started, 15 April 1996.

[9] J-MASS-GS-3.0a, J-MASS User Manuals - Volume IV - Model Developer Manual, Appendix A, 15 April 1996.

[10] Bezdek, W.B. and Reidelberger, E.A., "Vehicle Interactive Digital Pilot Model Using J-MASS", AIAA 96-1234, July, 1996.

[11] Schoening, W., "J-MASS for Systems Engineering", NCOSE 95-1234, July, 1995.

[12] Beavin, W., Emrich, T. and Bezdek, W., "J-MASS DIS Interface Using J-MASS", NAECON, May 1995.

AIAA-96-3493-CP

PRACTICAL LIMITS OF SUPERMANEUVERABILITY AND FULL ENVELOPE AGILITY

Brian A. Kish[*]
Wright Laboratory, Wright-Patterson AFB, OH 45433
David R. Mittlesteadt[†]
Aeronautical Systems Center, Wright-Patterson AFB, OH 45433
Götz Wunderlich[‡] and J. Matthias Tokar[§]
Industrieanlagen-Betriebsgesellschaft, D-8012 Ottobrunn, Germany
Terry Hooper[¶] and Robin Hare[#]
Centre for Defence Analysis (Air), Farnborough, Hampshire, GU14 6TD, United Kingdom
Hubert Duchatelle[**]
DGA, 75753 Paris Cédex 15, France
Patrick Le Blaye[††]
ONERA, 13661 Salon Cédex Air, France

Abstract

This study examined the tradeoffs among aircraft, missile, and avionics agility to achieve improved operational effectiveness. The *Arena* war simulation program was used to model a beyond-visual-range, many-on-many air battle and generate within-visual-range starting conditions. The *Air-to-Air System Performance Evaluation Model* was used to model 1v1 engagements at the *Arena*-derived starting conditions. Results from various flight trials and manned simulation were used to verify the simulation process. The most significant increase in operational effectiveness came from coupling missile agility with avionics agility. Aircraft agility had only a minimal contribution. Firings, exchange ratios, and losses were the metrics used to compare the technology areas.

The views expressed are those of the authors and are not necessarily those of the organizations they represent.

Introduction

Since 1986 France, Germany, the United Kingdom, and the United States have collaborated on a study with the following two objectives:

- Through analysis and simulation, determine whether supermaneuverability is operationally useful in future air combat scenarios.
- If operationally useful and technically feasible, determine the practical limits of supermaneuverability and full envelope agility.

Supermaneuverability is defined as very high levels of maneuverability and agility throughout the flight envelope of a fighter aircraft, especially beyond maximum lift. Agility is defined as the ability to change states rapidly with precision. Full envelope agility contains airframe, missile, and avionics attributes.

Previous analyses and manned simulations for close-in-combat (CIC), primarily emphasizing 1v1, have indicated substantial improvements in air combat effectiveness when supermaneuverability (in particular, post-stall technology) was incorporated in advanced fighter designs. Recognizing that air combat scenarios are likely characterized by rapid transition from beyond-visual-range (BVR) to CIC involving multiple aircraft, the effect of supermaneuverability technologies on the outcome of this type of engagement needs to be determined.

[*] Flying Qualities Engineer. Member, AIAA.
[†] Modeling and Simulation Engineer
[‡] Diplom-Ingenieur
[§] Diplom-Mathematiker
[¶] Senior Consultant
[#] Senior Analyst
[**] Ingénieur Civil
[††] Ingénieur

Simulation Strategy

The study began with manned simulation at Industrieanlagen-Betriebsgesellschaft (iABG), involving two piloted aircraft and three computer generated aircraft. Four pilots were trained on the baseline post-stall aircraft including avionics, weapons, and scenarios. After the training phase, the pilots jointly determined possible starting conditions (geometry, speed, weight, weapons load, number of runs, etc.). The purpose of manned simulation was to create a data base to develop a digital pilot reaction model. The application of a batch model was necessary in order to generate the number of runs needed to accomplish the project goals. The batch simulation strategy involved three different programs. First, the *Arena* war simulation program was used to model a beyond-visual-range, many-on-many air battle and generate within-visual-range starting conditions. Second, the *Air-to-Air System Performance Evaluation Model (AASPEM)* was used to model 1v1 engagements at the *Arena*-derived starting conditions. Third, the *Abductory Induction Mechanism (AIM™)* was used to link *AASPEM* results to *Arena* as depicted in Fig. 1.

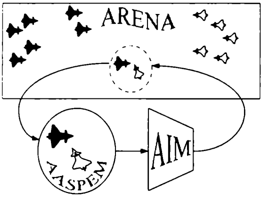


Fig. 1 Simulation interfaces

Arena was developed by the United Kingdom to investigate command, control, and communications; electronic countermeasures; and identify-friend-or-foe interactions in many-on-many BVR engagements. Aircraft and missile trajectories are modeled in detail up to the point where two aircraft enter CIC, as defined by the analyst. At this point *Arena* uses a statistical representation of CIC results to determine the outcome of the event. Statistics are created off-line using a separate model, in this study, *AASPEM*. If CIC criteria are fulfilled, the following steps are taken:

- Select appropriate CIC table
- Derive individual CIC result
- Return CIC outcome to *Arena* (losses, remaining a/c, etc.)
- *Arena* continues during CIC events.

Arena production runs were tailored to achieve various levels of CIC using the starting conditions below.

- 50 km separation
- 4 + 4 missile load
- 4 blue vs 4 red escorts + 8 red bombers
  - Limited CIC occurrence:
    - → No electronic countermeasures
    - → Vector to intercept
  - Some CIC occurrence:
    - → Reduced radar cross section
    - → Degraded air picture
    - → Reduced probability of kill given a hit at merge
  - Maximum CIC occurrence:
    - → Rule of engagement: Visual ID
    - → Medium altitude

AASPEM was developed in the United States to investigate air combat issues, from BVR through CIC. Although *AASPEM* can model BVR engagements, *Arena* is better suited for the complete air battle. *AASPEM* is limited to a total of 24 aircraft. Whereas, *Arena* is only limited by run time as the number of aircraft increases. *AASPEM* is a pseudo six degree-of-freedom simulator which better represents CIC. Eidetics International modified the code to incorporate enhanced fight maneuverability (EFM). This work included manned simulation with pilots from Germany, France, and the United States in two iterative campaigns. The following information is calculated and sent to *AIM™*:

- Missile launch / gun firing opportunities
- Missile launch / gun firing sequence
- Starting conditions (coordinates, speed, altitude, fuel, armament status, etc.) from *Arena*

AIM™, also developed in the United States, is a robust and compact transformation implemented as a layered network of feedforward functional elements. An example of a layered network is shown in Fig. 2.

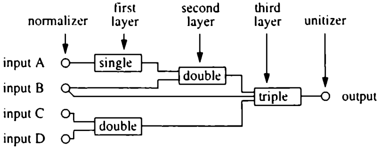


Fig. 2 Sample layered network

A formula can be fitted to the database created using results from *AASPEM*. The formula is then used in *Arena*.

Tactics

Three methods of modeling pilot tactics were used in this study. The first, called Novice, uses "look ahead" logic based on a strategy employed by many game playing computer models. The second, called Expert, is based on Novice. The third, called Roll and Pull, simply puts the lift vector in the line-of-sight plane to the target.

Novice Tactics. At a given time, the algorithm projects the "moves" or maneuvers an opponent can make. For each of these projected moves, the algorithm creates possible countermoves. The list of countermoves is then scored against a cost function, and the optimal move is selected. In CIC, such modeling takes on the following form:

Step 1 - Project Target Moves. At a given time increment, *dt*, each aircraft would project its target's position at a given future time, t_f. The lower bound on *dt* is the response time of the human nervous system. But *dt* is also increased to simulate the effects of pilot reaction delays. Such modeling features allow the user to investigate situational awareness effects.

Step 2 - Determine Ownship's Potential Maneuvers. Three fundamental parameters are used to change trajectory: pitch, roll, and thrust. It is possible, then, to generate trajectories from a given point by a combinatorial expansion. An example is shown below.

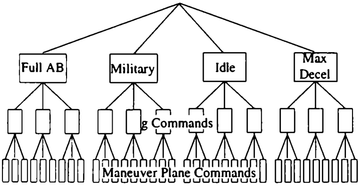


Fig. 3 Sample maneuver set expansion

At every maneuver decision point, the algorithm evaluates all possible maneuvers. The prediction is carried out to time t_f. Every one of these predicted moves is then scored against a cost function, and the best scoring trajectory is selected.

Step 3 - Cost Function. The cost function for this study was derived with the following assumption: *Tactics are driven by the envelopes of the weapons.* The cost function is a function of altitude, range to adversary, aircraft velocity vector magnitudes, aircraft velocity vector orientations, and missile characteristics.

Expert Tactics. The exhaustive search method Novice uses is computationally expensive (about 10 times longer than real time); however, the tree of maneuvers can be "pruned" by using "rules-of-thumb" or heuristics before all possible maneuvers are examined. For the sample maneuver set, there are five primary considerations in the selection of thrust.

1. Is ownship within an infrared weapon employment zone?
2. Is ownship above corner speed?
3. Relative altitude of the threat
4. Offensive or defensive posture
5. Recovery altitude

These five factors limit the number of probable thrust settings as a result of "expert" knowledge.

Roll and Pull Tactics. A third method was desired that complemented the "look ahead" results while further reducing computational time. The purpose of Roll and Pull is to produce a reasonable tactic that leads to a terminal maneuver designed to either cause or prevent a kill. Roll and Pull simply puts the lift vector in the line-of-sight plane to the target. Roll and Pull computation is 20 times faster than Expert and 60 times faster than Novice.

Calculations

In order to analyze the results, the calculation of the various parameters needs to be explained. Probability of kill, probability of survival, and exchange ratio are the key parameters.

Probability of kill (P_k). Two weapons were used: missiles and guns. The P_k of a missile given a hit was fixed at 0.8. P_{HIT} was the result of each simulated missile fly-out and on average was about 0.8. The P_k of a gun event was a function of the maximum burst length for one firing and the time duration of the gun hit. One burst, for example, fired five bullets in 0.2 seconds. The relation of gun P_k to bursts is shown below.

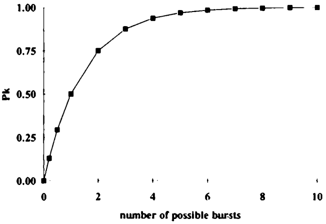


Fig. 4 Gun P_k

Probability of survival (P_s). The basic formula for calculating P_s for the blue aircraft is shown below.

$$P_{s_1}(\text{blue}) = P_{s_0}(\text{blue})\left[1 - P_{s_0}(\text{red}) \cdot P_{k_{event}}\right]$$

The following example is given to illustrate the calculation of P_s.

event 1 (20 seconds): blue fires a missile at red
event 2 (23 seconds): missile hits red (P_k=0.8)

$$P_s(\text{blue}) = 1$$

$$P_s(\text{red}) = 1[1 - 1(0.8)] = 0.2$$

event 3 (64 seconds): red fires a missile at blue
event 4 (65 seconds): blue fires a missile at red
event 5 (67 seconds): missile hits blue (P_k=0.8)
event 6 (68 seconds): missile hits red (P_k=0.8)

$$P_s(\text{blue}) = 1[1 - 0.2(0.8)] = 0.84$$

$$P_s(\text{red}) = 0.2[1 - 1(0.8)] = 0.04$$

Exchange Ratio: The following formula was used to calculate exchange ratio.

$$\text{Exchange Ratio} = \frac{[1 - P_s(\text{red})]}{[1 - P_s(\text{blue})]}$$

An exchange ratio of unity means the fight ended in a draw, values greater than unity favored blue, and values less than unity favored red.

Results

Model Tests

Four excursions were run to investigate *EFM-AASPEM* behavior and compare it with independent flight trials as well as manned simulations.

The first excursion examined the EFM flight trial starting conditions shown below.

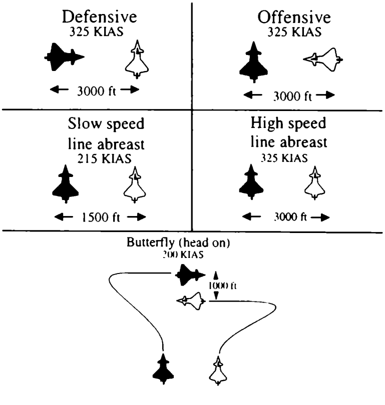


Fig 5. EFM flight test starting conditions

The aircraft were equipped with simplified AIM-9L type missiles and a simplified gun. The simplifications were based on the need to use "rule-of-thumb" weapon guidance, because the X-31 does not have suitable combat avionics. The following data were used for all *EFM-AASPEM* excursions:

- blue aircraft: enhanced pitch/torsional/axial agility, to represent X-31 type aircraft
- red aircraft: baseline aircraft, to represent F-18 type aircraft
- short range missile (SRM): baseline SRM to represent AIM-9L type missiles

For test purposes all three tactics available in *EFM-AASPEM* (Novice, Expert, and Roll & Pull) were applied.

Table 1 *EFM-AASPEM* tactics comparison

	SRM firings		gun firings			blue	red
Tactics	blue	red	blue	red	draw	wins	wins
Novice	0	2	36	2	3	21	1
Expert	0	0	30	2	3	20	2
R & P	0	0	34	0	3	22	0

The results indicate all three tactics achieve similar results. Comparing these results with flight test, it can be concluded that *EFM-AASPEM* generates the same trends as shown below.

Table 2 Comparison to flight test

	draw	blue wins	red wins	missile kill per engagement
Flight Test	6%	91%	3%	0.20
EFM-AASPEM	12%	82%	6%	0.10

The second excursion examined the trade-off between aircraft capability and short range missile capability. For this purpose, the blue aircraft was equipped with the baseline SRM. The red aircraft was equipped with the "best" SRM which had an increased angle-of-attack (AOA) limit, an increased off-boresight capability (OBC), and reduced minimum range (R_{min}). Starting conditions were the same as the first excursion.

Again all three *EFM-AASPEM* tactics were applied, and the following results were achieved:

Table 3 *EFM-AASPEM* trade-off results

	SRM firings		gun firings			blue	red
Tactics	blue	red	blue	red	draw	wins	wins
Novice	0	6	23	0	4	13	8
Expert	0	7	24	1	3	12	10
R & P	0	7	27	0	3	13	9

As before, the results indicate all three tactics achieve similar results. This excursion indicates the applied aircraft-SRM combination shifts the results significantly towards an enhanced missile effectiveness.

The third excursion examined SRM differences alone, by using the baseline aircraft on both sides. One aircraft was equipped with the baseline SRM while the other was equipped with the "best" SRM. No guns were included. This excursion was run with starting conditions used during manned combat simulations undertaken at iABG in 1993.

Table 4 Starting conditions

Engagement Aspect Angle	red 180° blue 0°	red 135 blue 45	
blue velocity (Mach)	Low 0.3	Med 0.5	High 0.9
red velocity (Mach)	Low 0.3	Med 0.5	High 0.9
co-altitude	5,000 ft	15,000 ft	40,000 ft

The results from *EFM-AASPEM* and the manned simulation are given below.

Table 5 Comparison to manned simulation

	EFM-AASPEM		Manned Simulation		
	firings/engagement		firings/engagement		exchange
Tactics	blue	red	blue	red	ratio
Novice	2.5	0	0.9	0.5	3.0
Expert	2.3	0.2			
R & P	1.3	0			

While Novice and Expert achieve similar results, Roll & Pull does not make full use of enhanced missile capability in very close starting conditions.

Although the numerical results (firings per engagement) for the *EFM-AASPEM* runs and manned simulations do not agree, it should to be noted that missile kills ended the engagement in manned simulation (with P_k up to 0.9). *EFM-AASPEM* continued for a fixed period. Missile enhancements are shown to be significant in both cases.

During manned simulation, a statistic involving range, OBC, and off-tail angle (OTA) at missile launch was generated to provide insight into the important issue of missile launch geometry. The *EFM-AASPEM* runs produced the following data:

Table 6 *EFM-AASPEM* launch geometries

OBC (°)			OTA (°)			Range (m)		
min	avg	max	min	avg	max	min	avg	max
0	50	70	0	115	180	750	2250	6000

In Fig. 6, the distance from the origin represents the range of the target from the launch aircraft. The angles of the center line represent the angle of the target aircraft off the nose of the launch aircraft. The dots show individual firings from manned simulations; the left hand side shows those which led to kills, while the right hand side shows misses. The corresponding results of successful *EFM-AASPEM* firings is shown as the shaded region.

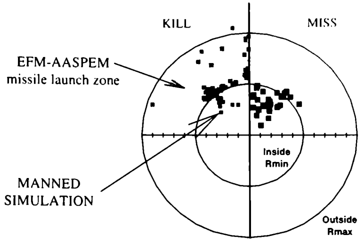


Fig. 6 OBC and relative distance at launch

Thus the comparison with manned simulation data (from the 1993 iABG trial mentioned above) indicated similar geometries in terms of OBC, OTA and range.

Within-Visual-Range Production Runs

The evaluation matrix for *EFM-AASPEM* (Table 8) was run against 40 starting conditions. The 20 starting conditions shown below were then reversed (red and blue) to form 40. Speeds ranged from Mach 0.8 to Mach 1.3. Altitudes ranged from 5,000 ft to 15,000 ft. These within-visual-range (WVR) starting conditions were generated from *Arena*.

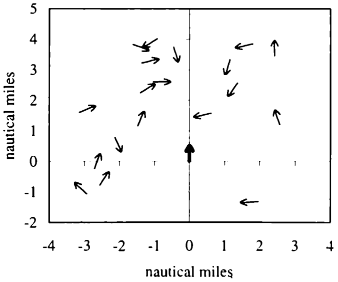


Fig. 7 WVR starting conditions

Rules of engagement. The following rules of engagement were used for 1v1 engagements:

- 120 second duration
- No kill removals

- Each aircraft started with same fuel load
- Each aircraft had 4 missiles and a gun
- Conditions for gun firing:
  - Minimum range = 500 ft
  - Maximum range = 3500 ft
  - Tracking delay = 0.2 seconds
  - Pipper size = 3.5 MIL (±2°)

The eight comparison areas in Table 7 were created using cases in Table 8. (A/C=aircraft, AV=enhanced avionics, BSL=baseline, OBS=off-boresight)

Table 7 Comparison areas

	Cases	A/C	SRM	AV	Comments
1	BL,A,B,C	•	BSL	BSL	Agility A/C
2	BL,F,G,H,I	•	BSL	BSL	Other A/C
3	BL,K,L	BSL	•	BSL	SRM
4	BL,S	BSL	BSL	•	AV
5	BL,M,Y	BSL	•	•	SRM+AV
6	BL,J,N	•	•	BSL	SRM+A/C
7	BL,U	•	BSL	•	AV+A/C
8	BL,D,O,Q,W	•	•	•	AV+SRM+A/C

1. Aircraft Agility Gun fire opportunities increased significantly by using enhanced-agility aircraft. The average number of gun fire opportunities increased by a factor of four.

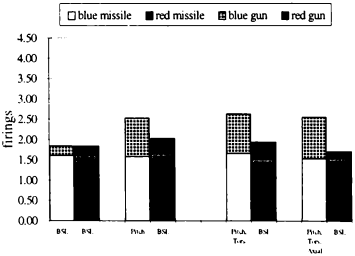


Fig. 8 Aircraft agility (firings)

Missile launches were, as expected, more or less steady around 1.5-1.6 missiles per engagement. Gun contribution to the total number of firings increased from about 15% (for the baseline) up to above 40% for the most agile aircraft. Because the missile P_k was 0.8 and the starting conditions were outside R_{min}, missiles contributed earlier and more to overall kills. This is reflected in exchange ratio, which shows a moderate increase of up to 10% in the case of the most agile aircraft, which is the result of the fourfold increase in gun firing opportunities.

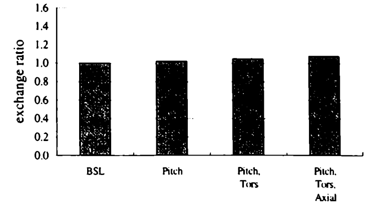


Fig. 9 Aircraft agility (exchange ratios)

Note that losses were high (around 45%) for the baseline case; aircraft agility increased the red losses by 20%, while blue losses increased slightly.

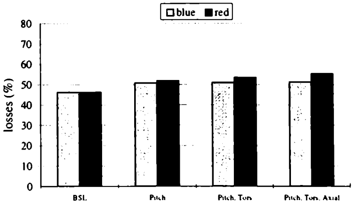


Fig 10. Aircraft agility (losses)

2. Other Aircraft Changes The enhancements "increased T/W" and "better wing efficiency" both achieved similar combat improvements to the enhanced agility aircraft (more or less constant levels of missile launches, but significantly increased levels of gun firing opportunities). The twofold degradation "decreased T/W with decreased wing efficiency" gave somewhat fewer missile launches and virtually no gunfire. This combination (labeled "deg" on Figs. 11-13) actually increased red missile and gun firing opportunities. Thus, "deg" had a *moderate* decrease in *offensive* aircraft potential, but a *significant* decrease in *defensive* potential. The other case of "degraded conventional aircraft performance coupled with best agility" was particularly interesting in terms of increased offensive potential (increased firing opportunities). However, added aircraft agility (corresponding to offensive capability) did not compensate for the loss of defensive potential (due to degraded T/W and wing efficiency) as seen by the increase in red firing opportunities. Exchange ratio shows slight improvement for the case of increased T/W (exchange ratio=1.1) and no improvement for increased wing efficiency. The other cases did significantly worse than the baseline aircraft.

In particular, enhanced agility did not compensate for conventional aircraft performance degradation.

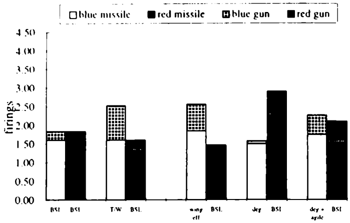


Fig. 11 Other aircraft changes (firings)

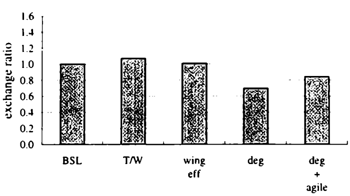


Fig 12. Other aircraft changes (exchange ratios)

Again, losses were high for the baseline aircraft (45%) and even higher for the degraded aircraft (65%).

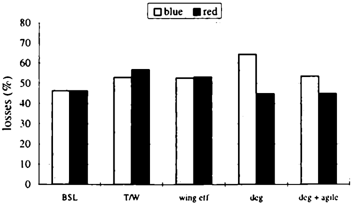


Fig 13. Other aircraft changes (losses)

3. Enhanced Missile Capability In the case of enhanced missile capability, gun firing opportunities were more or less the same, as expected. Increasing OBC raised the number of missile launch opportunities significantly by nearly twofold (from 1.6 to 2.9). Improved R_{min} added another 20% (to achieve a level of 3.5 launches per engagement). It should be mentioned the R_{min} improvement here was about 100% better (R_{min} halved) than the baseline SRM. Future SRM concepts will achieve further improvements (another 50% or more) which, coupled with high OBC, would further increase the value of R_{min} to operational effectiveness.

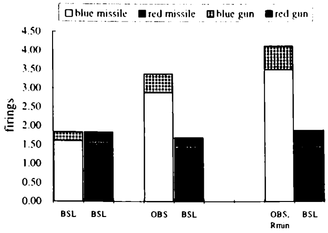


Fig. 14 Missile enhancements (firings)

Concerning exchange ratio in the *Arena*-derived scenario, enhanced OBC raised the exchange ratio to 1.3. With improved R_{min}, the exchange ratio rose to above 1.4. Both exceeded the effect of enhanced aircraft agility or performance (exchange ratios did not exceed 1.1). Thus, missile enhancements turned out to be attractive operational effectiveness drivers.

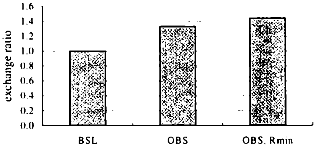


Fig. 15 Missile enhancements (exchange ratios)

Blue losses were high (around 45%). However, red losses increased drastically to near 70% against the most capable missile option.

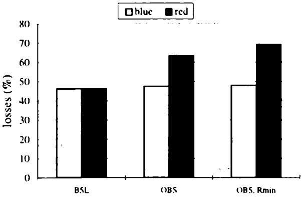


Fig. 16 Missile enhancements (losses)

4. Enhanced Avionics Enhanced avionics on the baseline aircraft with the baseline SRM did not pay off (as expected). The slight increase in red firing

opportunities indicates a mismatch of sensor-weapon capability, which sometimes resulted in misuse.

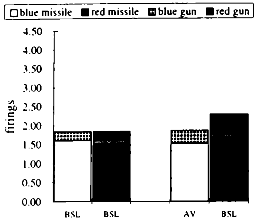


Fig. 17 Avionics enhancements (firings)

Overall, there was virtually no effect on exchange ratio or losses (which, once again, were around 45%).

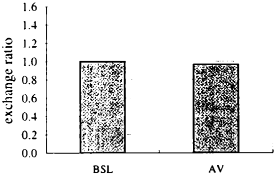


Fig. 18 Avionics enhancements (exchange ratios)

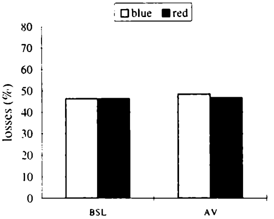


Fig. 19 Avionics enhancements (losses)

5. Combined SRM and Avionics Enhancements

Combined SRM and avionics improvements increased missile launch opportunities from 1.5 missiles per engagement to 2.3 with OBC improvement. Missile launch opportunities increased further to 3.3 when both OBC and R_{min} were improved . Additionally, gun firing opportunities increased with enhanced avionics.

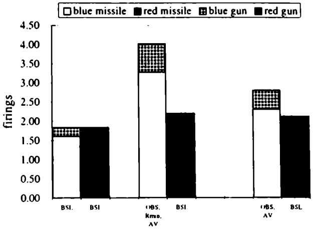


Fig. 20 SRM-avionics enhancements (firings)

Exchange ratio went up to 1.5 (10% beyond that for the same missile using baseline avionics).

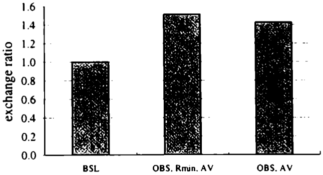


Fig. 21 SRM-avionics enhancements (exchange ratios)

Blue losses are again high (40-45%). Red losses were slightly lower than that for the same missile using baseline avionics, but still significant approaching 70%.

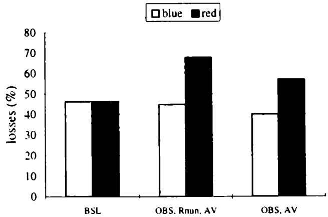


Fig. 22 SRM-avionics enhancements (losses)

6. Aircraft Agility with SRM Enhancements

Combining aircraft agility with SRM agility (using baseline avionics) increased blue firing opportunities. Missile launch opportunities were slightly lower than combining SRM and avionics enhancements, but the aircraft agility increased gunfire, as observed earlier. Removing the AOA limit for successful missile launch achieved only slight improvement in terms of missile launch opportunities. The largest improvement came from both enhanced OBC and reduced R_{min}.

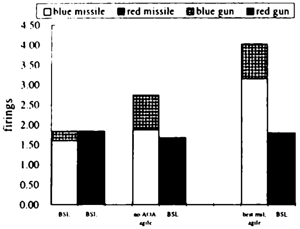


Fig. 23 A/C agility - SRM enhancements (firings)

The effect of both improvements (aircraft and missile) improved exchange ratios. Compared to the aircraft agility case (exchange ratio=1.1), removing the missile AOA limit gave an exchange ratio of 1.15. Improvements in OBC and R_{min} made the exchange ratio climb near 1.3 using baseline avionics.

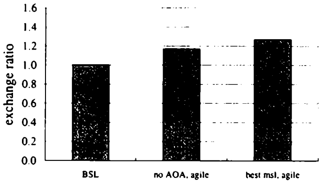


Fig. 24 A/C agility - SRM enhancements (exchange ratios)

Losses remained high (up to 52% for blue and up to 66% for red).

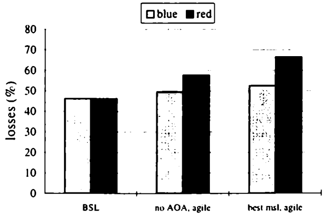


Fig. 25 A/C agility - SRM enhancements (losses)

7. Aircraft Agility with Avionics Enhancements As seen earlier, aircraft agility achieved a significant increase in gun firing opportunities (a factor of four) and a slight increase in exchange ratio (about 1.1). These grew with the addition of enhanced avionics. Missile launches increased, and gun firing opportunities increased to a factor of five.

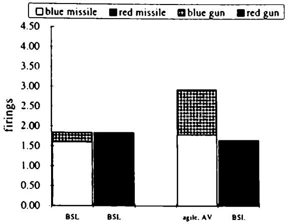


Fig. 26 A/C agility - AV enhancements (firings)

Both enhancements brought the exchange ratio up to 1.2, which indicates the additional impact of the enhanced avionics.

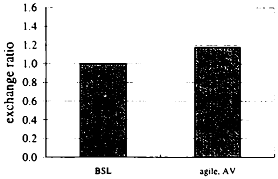


Fig. 27 A/C agility - AV enhancements (exchange ratios)

Red losses were again higher than blue losses (20% higher), but both remained high (above 45%).

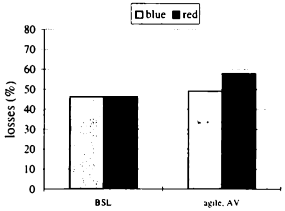


Fig. 28 A/C agility - AV enhancements (losses)

8. Combined Enhancements This comparison contains those cases which include modifications to all three subsystems: aircraft, missile, and avionics. One might expect the "best" system to be Case O with the "best" aircraft agility (pitch/torsional/axial), "best" SRM, and "best" avionics. Looking at firing opportunities, this was indeed true. Missile launches increased by a factor of two, and gun firings increased by a factor of three. Exchange ratio rose to about 1.4 (similar to enhanced missile capability alone; i.e. missile enhancements had the most impact in these *Arena*-derived scenarios).

One might expect the second "best" system to be Case D which is Case O but with degraded T/W and wing efficiency. Firings were slightly lower than Case O, but the exchange ratio was the highest of all cases (1.55). This effect was based on some single engagements where a "low-probability" event of firing multiple missiles in quick succession occurred. The P_k accumulation washed out all follow-on actions of the opponent. For example, two missiles (P_k=0.8) fired in quick succession resulted in a red P_s of 0.04.

One might expect the third "best" system to be Case W with "best" aircraft agility, "best" avionics, and a missile with improved OBC and R_{min} but with the AOA launch limit still imposed. The effect can be observed in firing opportunities. Slightly more gun firing opportunities partly compensated for the decreased number of missile launches. Exchange ratio decreased to 1.25 (compared to 1.4 for the "best" system above), but this exchange ratio was better than the agile aircraft with baseline missile and avionics (exchange ratio=1.1).

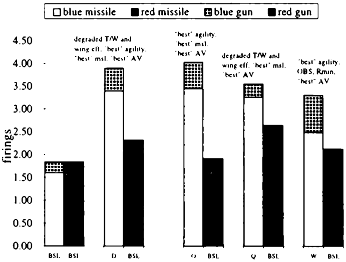


Fig. 29 Combined enhancements (firings)

Finally, a degraded aircraft with the "best" missile and "best" avionics was evaluated (Case Q). Missile launches showed a distinct improvement over the baseline case. Missile improvements contributed to the "offensive" part of the engagements in terms of success (increased red losses); however, as observed earlier, the decreased "defensive" capability of the degraded aircraft (degraded T/W and wing efficiency) enabled the red to increase the number of missile launches. Hence, there was no increase in exchange ratio.

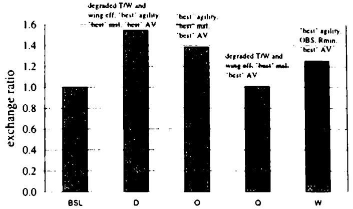


Fig. 30 Combined enhancements (exchange ratios)

Again, blue losses were high (48% for the "best" system). Red losses increased to beyond 70%.

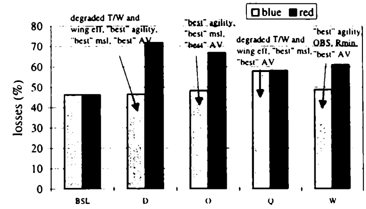


Fig. 31 Combined enhancements (losses)

Conclusions

Evaluation of 1v1 combat begins with starting conditions. Whether or not the starting conditions are realistic is a topic for debate. For example, is starting line abreast at 400 KIAS with a separation of 3000 ft a realistic starting condition? Often, the question "but how did they get there" is asked. It was decided to evaluate supermaneuverability and full envelope agility in context of a complete air battle. Thus, within-visual-range starting conditions were generated from the war simulation program *Arena*. Since *Arena* only uses three degrees-of-freedom, *EFM-AASPEM* was chosen to model 1v1 engagements using the *Arena*-derived starting conditions. *AIM*^{TM} was required to convert *EFM-AASPEM* results back to *Arena*. For this paper, two sets of results were given. First, four excursions were run to investigate *EFM-AASPEM* behavior compared to independent flight trials and manned simulations. Second, within-visual-range combat was examined against a number of technology combinations (Table 8) at 40 *Arena*-derived starting conditions.

Though close-in-combat was not a study item in itself, excursions were run to check if *EFM-AASPEM* could cope with this type of scenario. These excursions were successful and fully supported enhanced fighter maneuverability (EFM) flight trials as well as manned simulations concerning modern/future short-range missiles (SRMs). The conclusion therefore is EFM pays off significantly if air fights start very close in 1v1 situations inside the minimum range (R_{min}) of present-day SRMs with limited off-boresight capability (OBC), up to 30°, and no further aircraft entering the combat. EFM largely enhances gun firing opportunities. The frequency, how often those conditions will occur, is the subject of the *Arena* runs to be documented in the future.

The entire *EFM-AASPEM* work performed during the study was devoted to within-visual-range 1v1 combat. Comparisons were made based on firing opportunities, exchange ratio, and losses. The following conclusions were drawn:

- The most significant contribution to operational effectiveness was increased OBC coupled with enhanced avionics (mainly due to helmet mounted displays). Further improvements were possible when R_{min} was reduced.
- Missile and avionics enhancements have to be harmonized to fully make use of the improvement potential. It should be noted that missile/avionics OBC enhancements will provide even higher impacts in the many-on-many environment.
- Aircraft agility contributes to a certain extent, although not as significantly as missile/avionics enhancements. To make full use of agility, new aircraft designs might be required concerning aircraft kinematics and aerodynamics.
- Conventional aircraft performance enhancements do not improve system effectiveness. If envisaged, they would also require new aircraft designs.
- Degraded aircraft performance can hardly be compensated by enhanced agility. The degradation decreases the conventional turn capability which is a "defensive" potential. A decrease of this potential enables the opponent to generate increased firing opportunities.
- Degraded aircraft performance might be compensated by suitable missile/avionics enhancements. Although the same degradation concerning "defensive" potential applies, more firing opportunities can be generated earlier.
- Even taking into account that digital simulations tend to "over-rate", the losses were very high (above 40% during the simulation). This important result deserves special attention. While aircraft, missile, or avionics improvements may increase hostile losses, they ***DO NOT*** improve own survivability.

During the last 10-12 years, there has been significant improvement in missile technology. Next generation missiles make successful use of thrust vector control, thereby improving missile agility in the close-in environment as well as endgame performance. In addition, avionics have improved to make use of high OBC.

These developments make the new generation SRM/avionics attractive; however, the high mutual loss rates (expected to increase further) with all type of enhancements will "stress" the recommendation to urgently improve situational awareness as well as beyond-visual-range effectiveness to avoid WVR/CIC.

Table 8 Overall evaluation matrix

Case	BL	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Aircraft																											
Baseline	x											x	x	x						x	x					x	x
Increased Pitch Agility		x																									
Increased Pitch & Torsional Agility			x																								
Increased Pitch, Torsional, & Axial Agility				x							x				x	x	x					x	x	x	x		
Increased T/W							x																				
Better Wing Efficiency								x																			
Degraded T/W & Wing Eff									x									x	x								
Degraded T/W & Wing Eff + Best Agility					x	x				x																	
Missile																											
Baseline	x	x	x	x			x	x	x	x										x	x	x	x				
No Launch AOA Limit											x																
Increased OBS Capability												x														x	x
Increased OBS & Reduced Min Rng													x	x										x	x		
Increased OBS, Reduced Min Rng & No Launch AOA Limit					x	x									x	x	x	x	x								
Avionics																											
Baseline	x	x	x	x			x	x	x	x	x	x	x		x												
Enhanced 1					x									x		x		x		x		x		x		x	
Enhanced 2						x											x		x		x		x		x		x

AIAA-96-3494-CP

VEHICLE INTERACTIVE DIGITAL PILOT MODEL USING J-MASS

William J. Bezdek [*]
McDonnell Douglas Corporation
St. Louis, MO

Elizabeth Reidelberger
McDonnell Douglas Corporation
St. Louis, MO

ABSTRACT

There have been many digital pilot systems written in the past to provide digital control of aircraft, ground vehicles, or groups of objects. Traditionally these digital pilot models relied on hard coded algorithms that provided movement with limited interaction of the models which were used in the simulation. Other digital pilot models rely on "learning algorithms or genetic code" that use a weighting system and multiple runs to determine the optimum solution related to a combination of input conditions. The Joint Modeling and Simulation System (J-MASS) was used to help develop a digital pilot model which has unique capabilities derived from the object oriented port design of the model: can be configured using external data at run-time; and has the ability to take complete control of the vehicle, make interactive inputs to aid the operator, take partial control of some functions of the vehicle, or allow the operator to have complete control. This paper will describe the origin of the Digital Pilot designed/developed for J-MASS, discuss the development and use of different versions as well as lessons learned.

BACKGROUND

Digital pilot systems have been used for many years to provide an accurate and repeatable way of controlling aircraft, ships, ground vehicles, or groups of objects representing "real-world" battle forces (such as a carrier battle group, a wing of aircraft, a battalion, etc.). These software models, for the most part, rely on hard coded algorithms to control movement.[1 2] Later digital pilot models were developed which used a weighting system to determine the best maneuver to select for some time interval and provided the movement of the models used in the simulation.[3] Other digital pilot models rely on "learning algorithms", "genetic code" or a "pilot associate" that use a weighting system in conjunction with multiple runs to discover the optimum solution related to a combination of input conditions.[4 5] Older FORTRAN digital pilot models have been upgraded and rewritten in a more modern and realistic object oriented format in C++ or Ada.[6] These digital pilot models provide a basis for the implementation of a more complete, robust, object oriented, and realistic model to use in a variety of Joint Modeling and Simulation System (J-MASS) situations. A more detailed discussion of the various types of digital pilots, as well as an explanation of J-MASS, follows.

FLIGHT SIMULATION DIGITAL PILOT

For more than 15 years, McDonnell Douglas Aerospace (MDA) has used a digital pilot model which contains maneuver flags that are set by the operator / instructor to control the targets used in a manned simulation exercise. These maneuvers range from straight and level flight to an aggressive target intent to camp at the 6 o'clock position of the other aircraft.

The digital pilot is available as either an all Ada or FORTRAN model. The digital pilot generates the commands to the airframe control object which have been mapped to the pilot controls: stick, rudder, and throttle position. Digital pilot inputs include altitude, airspeed, specified load factor in a turn, inertial x, y, and z waypoint following commands, and relative target location.

Interactive control is available using the flight simulation control console to vary the maneuver number and the desired digital pilot conditions in real time. A skilled "Console Pilot" can manipulate the maneuver flags, load factor, altitude and position commands of the digital pilot to give realistic training for a manned pilot or provide a repeatable digital target for weapon system testing.

An upgrade to the digital pilot was developed for an advanced aircraft study that allowed the "Console Pilot" to control a series of aircraft

[*] Senior Member

functions using a graphical user interface (GUI) which allowed selection of an aircraft and control of multiple functions with the click of a mouse. For example, the digital pilot would close the aircraft to a specified range, fire AMRAAM missiles, beam (turn 90 degrees), then press the attack using Sidewinders and guns.

ADAPTIVE MANEUVERING LOGIC (AML) DIGITAL PILOT

The AML digital pilot used a weighting scheme to examine the predicted outcomes of a number of candidate maneuvers based on the predicted position and attitude of the target and the digital pilot. The candidate maneuver with the highest score was executed for the time frame until a new maneuver was chosen. The AML type of digital pilot has been used in several pilot trainers to provide a scaleable skill type of target.

An example of the use of the AML scheme was the Programmed Target, which was first installed on Device 2E6 Air Combat Maneuvering Simulator (ACMS) in 1979. The ACMS is a versatile fixed-based simulator for visual air-to-air combat training of F-4 and F-14 crew members, located at Naval Air Station (NAS) Oceana, Virginia. The targets, which can be used digitally or manned, include several threats as well as the F/A-18. The Programmed Target had five levels of operator / instructor selectable operation:

1. straight and level at commanded altitude and airspeed
2. sustained turns at commanded altitude, airspeed, and g-level
3. degraded mode
4. full-up mode
5. wingman mode.

Levels 3 and 4 made use of the AML approach, differing in how many g's the digital pilot was allowed to pull during a maneuver.

The Programmed Target could simulate any one of a number of selectable aircraft. The required data for each type of aircraft to be simulated was provided as data tables in the Programmed Target. This data included lift, drag, thrust, corner velocity, and recovery angle.

GENETIC ALGORITHMS FOR DIGITAL PILOT

A fairly new technique, which is applicable to all digital or manned simulations, is to allow the digital pilot to follow different flight paths and score the outcome after several runs. The maneuvers selected are based on the improvements obtained and recorded for different scenario set-ups. Normally, several hundred runs are needed to find an optimum solution. MDA has had significant success using genetic algorithms to determine optimum maneuvers. A recent test, using the X-31, used the genetics-based machine learning system to acquire rules for novel fighter combat maneuvers at high angle-of-attack. Using the Genetic Learning System (GLS) and a digital simulation model of 1V1 air combat, the GLS developed effective tactics for the X-31 using a comprehensive rule base for a large set of tactical maneuvers. Benefits of using this approach include:

- genetic classifiers can be directly converted into descriptive instructions for the pilot
- the classifier syntax not only describes the maneuver sequence, but also specifies the conditions relative to the target which triggers the controls
- the GLS can be run to accumulate rules from different sources including manned simulation
- variations in aircraft performance, starting conditions, etc. are transparent and require no changes to the software or procedures.

A novel feature of the GLS is the ability to learn against any opponent and represents a new application of machine learning for combat systems.

J-MASS

The Joint Modeling and Simulation System (J-MASS) is a government owned modeling and simulation system used to build, test, and perform analysis on simulation models. J-MASS is sponsored by the J-MASS program office at Wright Patterson AFB, Dayton, Ohio.

J-MASS is an object oriented, graphical modeling and simulation (M&S) system used to develop and simulate aircraft, missiles, radars, and a variety of models including digital pilots. J-MASS provides a wide range of modeling and simulation (M&S) capabilities which will include a/an:

- real world object oriented design methodology
- graphical programming environment
- software structural model that supports reuse using an automatic code generator for Ada or C++
- graphical configure capability to support scenario setup
- event based or real time processing environment
- distributed heterogeneous network for execution
- graphical execution monitor
- post-processing analysis and playback capability
- DIS compatible
- library of verified / validated model entities and components

- user manuals, training, and operational support
- real world terrain
- Sun and SGI workstation support

The J-MASS system is shown in Figure 1.

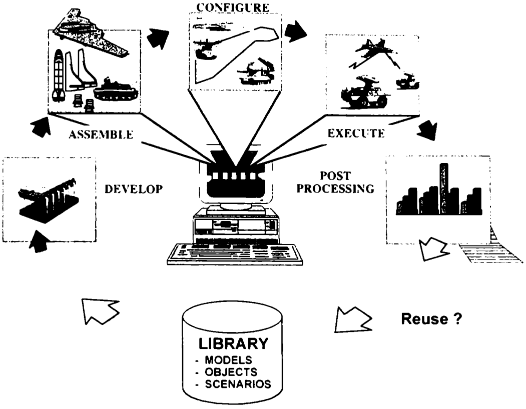


Figure 1 J-MASS System Concept

J-MASS supports few-on-few engineering analysis and simulation with varying levels of detail from analytic through emulative. The current release, JMASS 3.0a, includes a multi-target tutorial, containing the step-by-step procedure to set up, run, and analyze the simulation.⁷ Using the combination of graphical tools and the M&S system available with J-MASS, a simulation can quickly and easily be set-up similar to the target tutorial which has two targets following waypoints until they collide and a third target following a different path which has a close approach to the other targets without a collision. The scenario shown on Figure 2 will be used to test the capabilities of the digital pilot used in J-MASS.

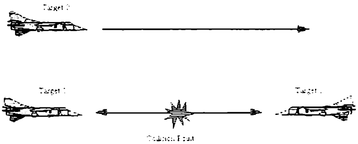


Figure 2 Target Scenario

GATM

The Generic Aircraft Target Model (GATM) is an object-oriented model that is written in C++. GATM provides a target that can be programmed to be an aircraft of many different types: from a cruise missile to a B-2 bomber. The system was designed using the Missile Space Intelligence Center C++ (MSIC++) environment. The basic capabilities of the GATM model are shown on Figure 3.

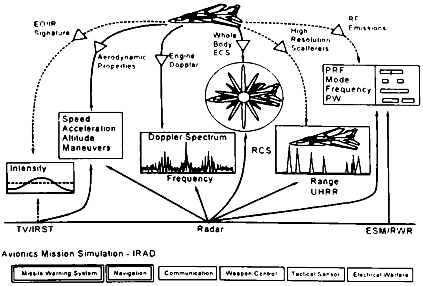


Figure 3 GATM Capabilities

The pilot object model uses maneuver commands, load factor, altitude, and waypoint commands to follow its flight path. The route is defined in terms of a starting location, heading, airspeed, and list of waypoints. Tasks to be performed during the mission may be maneuvers, triggers, or configuration changes; like wing sweep, stores release, etc. All tasks must have a trigger and several tasks may share the same trigger. A task may be triggered based on simulation time, a specific waypoint, or a missile detection. Using the mission planning file, the user can set up a realistic flight path with a wide variation in the tasks that can be performed.

The pilot object receives navigational data from the navigation object and flies the generic aircraft toward the next waypoint. At each waypoint, an event is triggered to select the next waypoint unless an event is received by the pilot object to alter the selection including:

- a missile warning event which starts a missile evasion
- a trigger for a specific maneuver
- no new maneuver which starts a cruise maneuver.

The pilot object is shown on Figure 4.

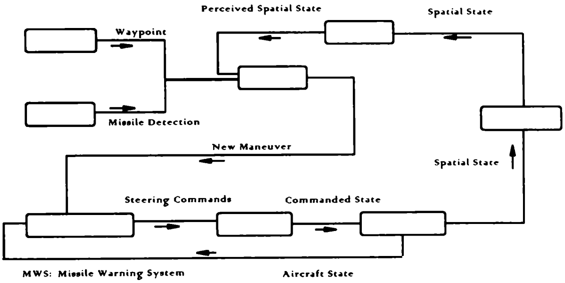


Figure 4 GATM Object Diagram

USING J-MASS TO DEVELOP A DIGITAL PILOT

J-MASS can be used effectively to design, develop, and simulate various objects, such as airplanes and missiles. J-MASS was used to design a complete, robust, object oriented, realistic digital pilot model to use in conjunction with air target models in a J-MASS simulation because of its extensive graphical and object-oriented capabilities. This digital pilot uses J-MASS tools to graphically configure the maneuvers, waypoints, etc.

J-MASS capabilities related to the development and use of a digital pilot model include the following:

- a generic target model using the graphical J-MASS development tools (Figure 5)
- a Digital Pilot designed/developed for J-MASS using ports to link to a generic target model
- modifying existing models
- building complete new models
- assigning attributes with default values that can be modified at runtime
- using the J-MASS automated code generator to modify or create new source code
- adding behavior code to define the generic algorithms
- compiling the model components
- combining the models into teams for execution
- configuring the data for the specific version of the model desired (Figure 6)
- configuring the models into a simulation using graphical scenario tools (Figure 7)
- viewing the simulation during execution using the Combat Battle Monitor (CBM) (Figure 8)
- journalizing the data for further analysis post-processing the results (Figure 9).

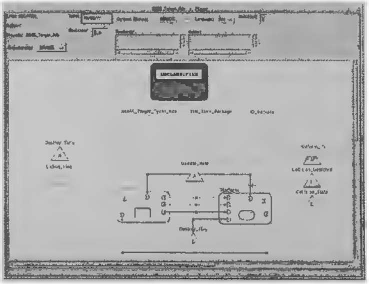


Figure 5 Generic JMASS 3.0a Target Model

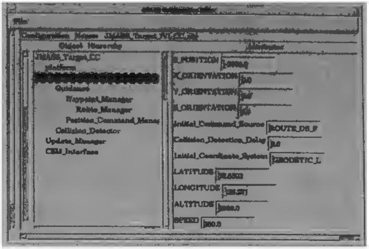


Figure 6 Configure Attribute Data

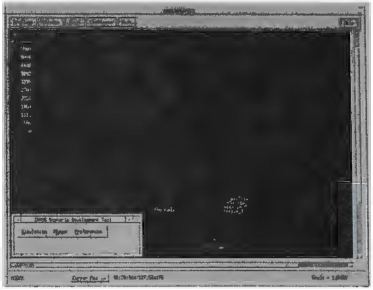


Figure 7 Typical Configure Scenario

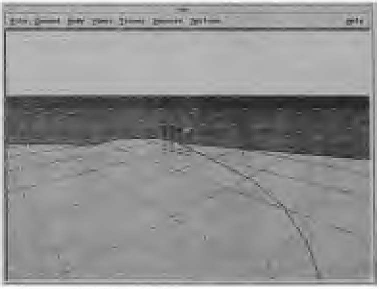


Figure 8 Typical CBM Display

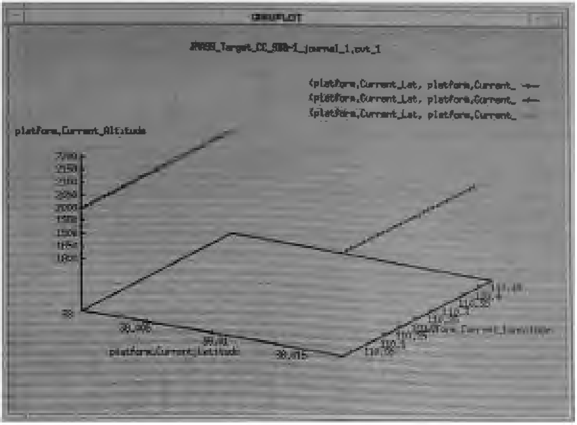


Figure 9 Post-process GNU Plot

The Digital Pilot designed / developed for J-MASS provides the digital pilot functions needed for a/an:

- air-to-air target
- waypoint following target aircraft
- air-to-ground weapon delivery following a specified route
- air-to-air intercept
- interactive aircraft that allows the user to provide control inputs
- manned aircraft controlled through stick and throttle input.

Unique areas of the Digital Pilot designed / developed for J-MASS model concept are:

- external J-MASS data driven configuration capability
- an object oriented port design
- digital, interactive, and manned capability.

MCDT USED TO LAYOUT DIGITAL PILOT

J-MASS has a unique graphical development tool called the Model Component Development Tool (MCDT) which was used to develop the digital pilot model. The MCDT allows the engineer / programmer to develop interfaces at a higher level using graphical tools and take advantage of an auto code generator to assist in writing the software. All J-MASS models use the same software structural model and an auto-code generator to provide code that is highly reusable and expandable for other functions. The data file for the digital pilot is external to the algorithmic (behavior) code, so the user can change the configuration and control inputs to the digital pilot without the need to recompile or reload the simulation. The basic data driven structure for the digital pilot is shown on Figure 10.

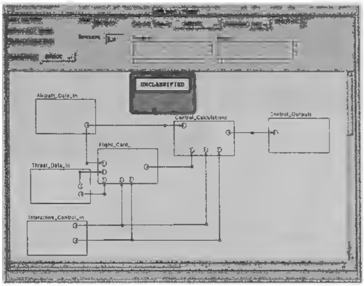


Figure 10 Digital Pilot MCDT

The Digital Pilot designed/developed for J-MASS was built as a separate object from the vehicle and interfaced to the vehicle using ports which may be on the same processor or across a network.[8]

DIGITAL PILOT CAPABILITIES

Several capabilities of other digital pilot models have been combined in the Digital Pilot designed / developed for J-MASS. The digital pilot can have complete control of the vehicle in the "all digital" mode, the operator can take partial control in "interactive" mode, or the operator can take full control in the "manned" mode. The operator can select the method of communication to be used through the keyboard, through a graphical interface, or through a control stick and throttle.

ALL DIGITAL CAPABILITY

The digital pilot provides stick and throttle inputs to the model, which are generated based on the digital calculations and the commanded maneuvers including:

- level flight
- sustained turns
- waypoints following

The interface to the digital pilot model is compliant with the J-MASS Application Program Interface (API). [9] Ports are used which interface the outputs of the digital pilot model to the inputs of the vehicle being simulated.

Basic maneuvers are selectable as a function of time, target, or cooperative vehicle maneuvers. A series of waypoints can be selected in configure mode, as shown in Figure 11, to define the desired path for the aircraft.

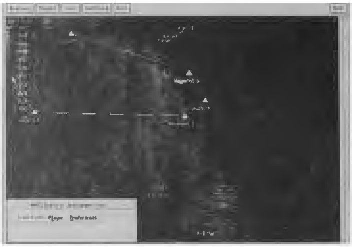


Figure 11 Digital Pilot Route

INTERACTIVE CAPABILITY

In addition to providing an all digital mode for use in J-MASS simulations, it is desirable to allow the user to interact with the models directly or at least to provide human inputs using the maneuvers available to the Digital Pilot designed/developed for J-MASS. This is not a new concept and has been used in trainers and flight simulators since the 1970s. The Programmed Target which was used on Device 2E6 Air Combat Maneuvering Simulator, and discussed earlier in this paper, would react immediately to the maneuver requested by the operator. Controls were provided on the Instructor/Operator Console to allow easy and fast maneuver selection by the operator.

To be effective in the interactive mode, the user must be able to observe the flight of the digital pilot either on a situational display or from an out-the-window (OTW) display, or both as was shown on Figure 8, Page 5. J-MASS provides this capability with the Combat Battle Monitor (CBM). CBM is a graphical tool for execution monitoring and replay. It has been modified to provide an interactive control panel to provide inputs to the Digital Pilot designed / developed for J-MASS.

CBM provides a graphical display of the aircraft and other elements of the simulation. It has a playback and data capture mode that is useful for data analysis, debriefing, and training of pilots. Interaction with CBM is through the mouse and keyboard. The main CBM format displays a scaleable background color map with selectable grid lines. It shows a Gods-Eye 2-D view or a 3-D perspective view as was shown on Figure 12. The user can select the out-the-window (OTW) view of the desired aircraft using CBM to observe the situation from the digital pilot's perspective. The user can use the information obtained from this view to make inputs to control the digital pilot.

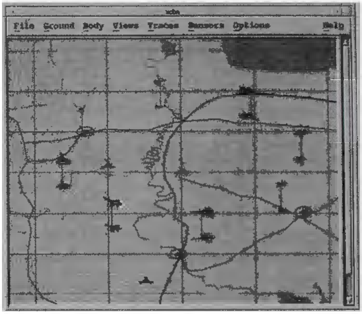


Figure 12 CBM Supports Pilot View

Air and ground targets are overlaid on the map in their correlated positions. They are color coded red for enemy targets and green for friendlies. Traces and altitude poles are selectable for the air targets. Flyouts of in-coming and out-going missiles are also shown on the display. In addition to the graphics window of the world, there are several data information panels available from pull down menus. Selecting an aircraft on the map with the mouse will put that symbol's data in the primary aircraft data field of the data panel, highlight that symbol on the map and re-center the map around that symbol. A grid may be turned on and spacing chosen. The Gods-Eye, 3-D mode, or pilot's view selection are found on this menu. Various flags are available to turn tracers, ribbons, and altitude poles on or off. Zooming of the map can be done from this panel or with the keyboard.

Inputs are provided to CBM from ports in the models which communicate from the targets and other players. A replay mode is available to select the CBM inputs to be saved for later replay and analysis.

The Digital Aircraft Control (DAC) panel on CBM is used to allow the user to control the target and the weapon system. The DAC panel was developed to provide a graphical user interface for controlling multiple digital aircraft within a simulation as shown on Figure 13 . Maneuvers and weapons are selectable by the user as shown on Figure 14.

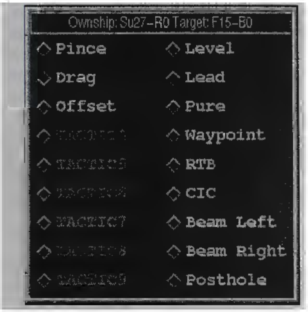


Figure 13 CBM Tactics Panel

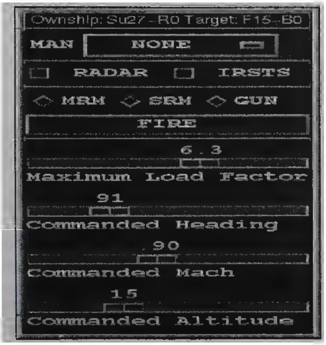


Figure 14 GCI Station

A similar panel has been started to provide interactive control of the aircraft to attack specified targets, perform cooperative tactics, provide stick, and throttle inputs. Networking DAC into a simulation enhances digital aircraft performance by giving more realistic behavior than current "digital only" models provide.

MANNED CAPABILITY

The graphical initialization capability available in J-MASS will be utilized for manned simulations which provides a "user friendly" way to initialize all elements of a simulation anywhere in the world and provide common file formats between different levels of simulation. The displays are the same ones used to initialize digital or interactive assets.

The OTW view on the CBM can be used to provide a cockpit view which allows the user to take either partial or complete control of the vehicle. The digital pilot allows user inputs in lieu of the digital or interactive control of the digital pilot to control the vehicle in the simulation. MDA has developed lower cost simulators including a control stick and throttle to provide full interactive control using an RS-232 interface in conjunction with the Digital Pilot designed/developed for J-MASS.

INCORPORATING CAPABILITIES INTO THE DIGITAL PILOT DESIGNED DEVELOPED FOR J-MASS

The digital pilot model requires a lot of flexibility to perform maneuvers and conform to real world usage. To provide this type of capability, a digital "Flight Card" model has been developed as part of the digital pilot model that was shown on Figure 10, page 6. The Flight Card MCDT is shown on Figure 15 .

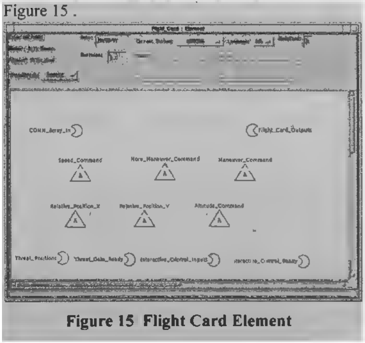


Figure 15 Flight Card Element

Digital pilot outputs are provided to J-MASS ports which allow the digital outputs to be used as inputs to the aircraft model or replaced with manned inputs without having to recompile since they both use stick and throttle positions. This allows the engineer to test the model with repeatable digital inputs, especially during development and then replace or supplement these inputs with interactive / pilot inputs as desired as shown on Figure 16.

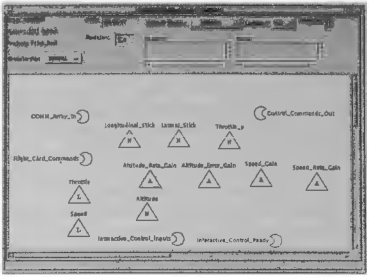


Figure 16 Digital Pilot Outputs

The Digital Pilot design/developed for J-MASS can be set-up at runtime including autopilot gains, desired modes, and calibration points as shown on Figure 17. This provides an easy way to modify the responsiveness of the control inputs without having to relink.

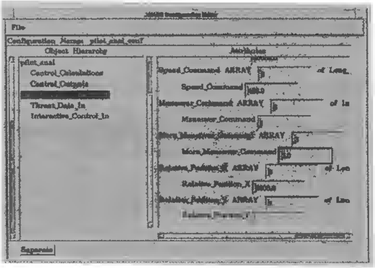


Figure 17 Digital Pilot Configuration

TEST WITH DIGITAL PILOT DESIGNED/DEVELOPED FOR J-MASS

The same digital pilot maneuvers used for the J-MASS tutorial were selected to test the Digital Pilot designed/developed for J-MASS. The platform (aircraft) model needed to be replaced to provide a separate player to interact with the digital pilot as shown on Figure 18. Aircraft modeling in J-MASS was discussed previously in a previous paper. [10]

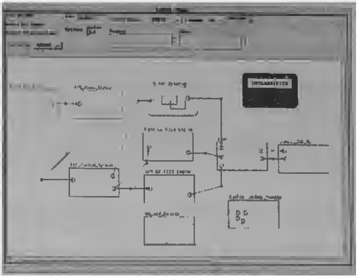


Figure 18 Aircraft Platform

DIGITAL TESTS - The maneuvers selected allow the digital pilot to have each aircraft perform a half circle and pass close enough to trigger the collision flag. The third aircraft was commanded to fly straight and level. The results are shown on Figure 19. This is the same result obtained using the J-MASS tutorials as discussed earlier.

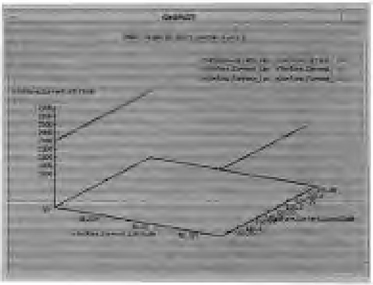


Figure 19 GNU Plot of Digital Run

INTERACTIVE/MANNED TESTS - Testing using the interactive and manned capabilities of the Digital Pilot designed/developed for J-MASS were deferred until a later time. The digital pilot software uses the same interface as provided by a manned simulation, so the maneuvers and stick and throttle inputs received provide realistic movement, since they interact with all the real world objects of the aircraft model. The type of stick and throttle planned for this test uses CBM to provide the OTW display and the DAC panel to provide interactive inputs. The stick and throttle were used from an existing simulation.

CONCLUSION

The purpose of the Digital Pilot designed / developed for J-MASS is to provide a digital pilot that supports digital, interactive, and manned control of single or aggregate groups of vehicles. One of the key features of the digital pilot is the ability to use it for a variety of different platforms and that the outputs are the same as the ones provided by a manned simulator. This system takes advantage of legacy code provided by other digital pilots including:

- digital pilots with maneuver commands to change state
- adaptive digital pilots which uses the expected outcome to select maneuvers
- genetic algorithms to help develop optimal pilot maneuvers
- digital pilots which follow a prescribed flight path, but can interact to specific events along the way
- interactive and manned pilot models which provide control inputs to the vehicle.

There are several advantages using J-MASS as the modeling and simulation environment for the Digital Pilot designed / developed for J-MASS including:

- graphical tools to build object oriented models
- automatic code generators to support Ada and C++ models with the same block diagram
- a library of objects and complete models in the local modeling library like aircraft, radars, missiles
- a configure capability so the models can be easily modified by the user without having to recompile
- a graphical configuration mode to place models into a scenario anywhere in the world
- graphical execution and post-processing tools to monitor simulation results
- an distributed network to allow the digital pilot to be separated on a different computer, network or cockpit
- a library of verified and validated models placed in a government reuse library
- customer support, training and user manuals
- models developed in J-MASS can be exported and used in other simulations

Using the modeling and simulation techniques of J-MASS, a digital pilot was designed / developed for J-MASS which provides an all digital, interactive, or manned simulation model. It is suitable for low level analysis to control individual aircraft or aggregate groups of aircraft, with dynamic simulation models often used in real time manned simulation, as a reference for the development of digital models which control other vehicles, or a model to be used in detailed simulations where significant control of the model is needed . In addition, the design supports expansion of the existing digital pilot to control groups of aircraft or other vehicles.

FUTURE WORK

The use of the Digital Pilot designed/developed for J-MASS was used only with aircraft models. The same basic concepts can be expanded to provide digital / interactive / manned control for a SAM site, a radar, a group of tanks/trucks or commander level: command, control, communications, computers and intelligence (C4I) functions. The interactions at different levels of command are very important for large scenarios.

The same capabilities of J-MASS could be applied using a low cost stick and throttle to provide the same RS-232 inputs to the computer system. With a low cost stick and throttle, each designer has at their desktop the basic controls, displays, avionics, aerodynamics that make up the vehicle under design or modification, i.e. an interoperable desktop design and development station. This would make the development of a new weapon system or upgrades to existing systems easier, provide improved use of an integrated product team, and reduce development costs.

With the combination of a J-MASS system to provide object oriented real-world model development, the configure capabilities of J-MASS to set-up realistic scenarios, a cockpit controller, cockpit display panels, and the appropriate computer equipment; an engineer can have considerable capability to design and test new concepts and / or interactively control a relatively large scenario.

REFERENCES

[1] Bezdek, W.B. and Galloway, T., "Digital Test Pilot Concepts", AIAA 72-1234, June, 1982.

[2] Stephenson, R., Emrich, T. & Jones, J., "Real Time Aircraft Simulation Using J-MASS", NAECON, May 1995.

[3] Adaptive Maneuvering Logic (AML) Users Manual, 1977.

[4] Handbook of Evolutionary Computations, Smith, R.E. & Dike, B.A., An Application of Genetic Algorithms to Air Combat Maneuvering, Oxford Press, 1996.

[5] Pilot Associate, Dr. Jim Guffy, 1987.

[6] MDC 94P0019, "Generic Aircraft Target Model Requirements and Development Manual - Software User Manual", Vol. 5, 30 November 1995.

[7] J-MASS-GS-3.0a, J-MASS User Manual - Volume II Getting Started, 15 April 1996.

[8] Beavin, W., Emrich, T. and Bezdek, W., "J-MASS DIS Interface Using J-MASS", NAECON, May 1995.

[9] J-MASS-MD-3.0a, J-MASS User Manuals - Volume IV - Model Developer Manual, Appendix A, 15 April 1996.

[10] Bezdek, W.B., "Aircraft Modeling and Simulation using J-MASS", AIAA 94-1234, June, 1994.

AIAA-96-3496-CP

USE OF A JOINT MULTI-FACILITY TRAINER FOR INTERNATIONAL SPACE STATION TRAINING

Joseph Policella, Senior Software Engineer
Hughes Training, Inc., Houston, Texas 77058
and
Stanley Allen, Senior Software Engineer
Hughes Training, Inc., Houston, Texas 77058
and
Harold Smith, Chief Engineer
Hughes Training, Inc., Houston, Texas 77058

The training of astronauts and ground crews for the National Aeronautics and Space Administration occurs at the Johnson Space Center. Traditionally, the simulators built to accomplish this training only trained teams at the Johnson Space Center site. Crew training of Space Station crews, including International Partner crews, will be distributed among their respective space centers using local high fidelity trainers. The International Partners' space centers are located in areas such as Japan, Germany, and Russia. To achieve combined team training with the geographically distributed simulations, all of the trainers must be combined into a Joint Multi-Facility Trainer or "virtual simulation." A virtual simulation allows time homogenous state updates on distributed trainers as well as correlation of distributed simulation databases in real-time. Due to the geographic location of the virtual simulation components, substantial transport delays will occur while data is transferred among the components. Also, changing requirements and technology upgrades can pose significant challenges to a virtual simulation. By adopting a common set of standards and Object-Oriented techniques, these problems can be alleviated or minimized. This presentation provides disclosure of a Space Station virtual simulation concept and associated standards to allow trainers on dissimilar platforms to participate in one virtual simulation.

Introduction

The National Aeronautics and Space Administration's (NASA) Johnson Space Center (JSC) has a long history of real-time simulator development for training. Simulators have been used as far back as the early 1960s on programs such as Mercury, Apollo, and Skylab. Recently, simulators have trained astronauts, flight crews, and ground controllers on the Spacelab, and Shuttle programs. Training has included every aspect of mission operations from launch and landing to in-orbit robotic arm manipulation. These simulators have encompassed a wide range of training devices from small part task trainers to fully integrated motion based trainers.

Distributed simulation has also played a role at JSC. On the Apollo program a Joint Multi-Facility Trainer, composed of elements from the Kennedy Space Center (KSC) and JSC facilities, was used to train Apollo launch and orbital insertion. Also, integration of simulators between JSC and the Marshall Space Flight Center (MSFC) was performed to train combined Spacelab systems and experiment training.

Although considered complex for their time, these Joint Multi-Facility Trainers pale in comparison to what is required of today's simulators. The levels of integration involved with currently proposed military, space, and commercial simulation projects require unprecedented global distribution, data exchange, and database correlation among the distributed simulators. Also, the ability to dynamically reconfigure a simulation requires precision in timing and coordination of the simulation components so that each simulation element can respond to the corresponding state change in a time homogenous manner.

To accomplish this, a common infrastructure must be developed to implement the virtual simulation concepts. This infrastructure must be designed with enough flexibility to provide End-To-End (ETE) virtual simulations for training, testing, and verification of:

- Advanced Systems
- Assembly Operations Rehearsal
- Payload Systems
- Whole System Operations
- Flight Crew/Ground Team Development
- Operational Procedures

This paper describes the concept of virtual simulation, how it was implemented at Hughes Training Inc. (HTI), and its application to the Space Station and Training Facility (SSTF) at JSC and international partner facilities.

Virtual Simulation

Concept

Simulations may be developed throughout a project/product lifecycle for a variety of purposes, including testing, concept evaluation, training, and quality assurance. In large system projects, these simulations (or *models*) are often created individually for pieces of the system(s), without a regard to joining them together for an end-to-end test-bed for simulation. In most cases, the behavior of an object in the real-world system is subject to a variety of stresses and state conditions which are difficult or expensive to account for in simulation unless the model for that object is part of a complete system simulation environment. Often, these stresses and state conditions are the result of real-time operation, which may be considered unnecessary to simulate for individual object models. Furthermore, the complete behavior of the larger system in its environment is often not comprehended until the system is deployed, after which redesigns and corrections become much more difficult and expensive.

In response to these difficulties, one approach may be to collect all the models for the "pieces" of the system and join them together in an *ad hoc* "multi-model" configuration. This has the benefit of retaining the investment made in the models of individual objects. However, this is often not feasible because of the incompatible nature of the different models, the expense of creating the "glue" for the models to coordinate and communicate, and the lack of an infrastructure which allows complete overview and control for the system simulation. Thus, in most large system projects, multiple models are generally not operated in conjunction for a "synergistic" effect, except on a small scale with only few models.

What is needed is a front-end approach to simulating the larger system and its environment, with the ability to incorporate object models in a systematic way, coordinate them and permit communication, and control the entire simulation as a whole. This is the concept we call *virtual simulation*, so called because the entire system is being simulated not by a single model, but by a collection of models coordinated to produce the virtual effect of a single, coherent system. An infrastructure called the Simulation Virtual Machine (SVM) has been developed by HTI which is used as a basis for implementing this concept in the SSTF.

Virtual simulation, for the purposes of this paper, is identified by the following criteria:

- Distributed (network-based) real-time operation.
- Common infrastructure for operation.
- Common coordination and communication architecture.
- Ability to incorporate previously developed models.
- Ability to tolerate multiple levels of model fidelity.
- Ability to incorporate real objects (hardware) without redesign.

Assuming that object models and their respective databases have been developed, virtual simulation is achieved by defining interfaces between the models in the context of the infrastructure's communication architecture, integrating the models into the simulation infrastructure to create a simulation element, and combining and reconfiguring distributed elements to meet the needs of a particular simulation session.

Implementation

At HTI, the virtual simulation implementation consists of (see Figure 1):

- The Simulation Virtual Machine (SVM).
- A simulation host platform.

- A variety of distributed computing platforms with object models or hardware elements.
- A network to link the platforms.

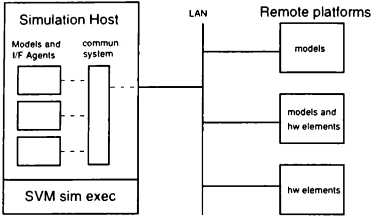


Figure 1: Virtual Simulation Implementation

SVM is a software layer which provides the common infrastructure for incorporating multiple models into a single system. n the virtual simulation environment, SVM executes models in the simulation host in real-time, and provides communication mechanisms for models and other services (time homogeneity of data, status and control capabilities, moding, and others).

Key to the incorporation of distributed platforms is the concept of an *interface agent*. An interface agent is a model in the simulation host platform which represents a remote platform (which may contain an object model or hardware element, or mix of both). The interface agent acts as a low fidelity version of a remote platform when that platform is not incorporated into the simulation session. When the remote platform is incorporated into the session, the interface agent provides a pass-through capability so that models on the simulation host are not concerned with whether real hardware or a software model is on the other end of the communication path.

Simulation Virtual Machine

SVM provides the infrastructure software for implementing the virtual simulation for SSTF. The executive is based on the concept of rate-monotonic scheduling, which permits models to execute in real-time without "frame-based" (manually coordinated) scheduling. Each model contains an SVM executive element (the "thread executive") which is executed as an Ada task. All such tasks are coordinated by a master task which is hidden from the model. Moding and control commands are passed into the model through the master task.

The only connection that a model has to other models is through Ada package specifications which define the data to be transmitted and received over the communications architecture, which is known as the software backplane (SWBP). The SWBP shields the model such that it is not concerned with the location or configuration of the source or sink, whether it is a model in the simulation host, an interface agent, or a remote platform. It keeps track of the location model and the data owned by each model. The SWBP also provides strict time-homogeneity for messaging so that a model receives data which is correct for its rate of execution. SVM provides timing synchronization for all platforms in a simulation session.

SVM provides templates for interface agents, which can be tailored for each instance. The template provides the communication facilities and coordination hooks into SVM to permit the interface agent to act as a stand-alone model when the remote asset it represents is off-line, or to pass-through data when the agent is on-line.

A GUI interface permits a session operator to easily control one or more simulation sessions. A simulation session is an object which permits the operations: create, terminate, add and drop of hardware elements, change modes, save and restore system state (datastore/reset points), and abort. The operator can examine and update model data, and insert malfunction conditions, using symbolic names defined by the model developer. There are tools which gather data at high rates, plot data on a graph relative to a time-line, and maintain scripts of control and data operations for repeated use.

SVM is designed for platform independence, and has been ported to a variety of machine architectures. This allows future asset development to take advantage of the latest technologies without sacrificing compatibility.

Space Station Simulation

Space Station Overview

The International Space Station will be the worlds largest flying vehicle. The Space Station will be assembled in orbit over a period of 5 years in states. During assembly until final completion, the Space

Station will have 44 different configurations. The vehicle will be supported by self contained power generation and life support systems, guidance navigation and control systems, thermal control systems, communications and tracking, and a multi-tiered network of on-board computer systems. The Command and Data Handling (C&DH) system consists of 48 space hardened Intel 80386 based computers, connected throughout the modules of the Space Station by MIL-STD 1553 busses. The C&DH system's main function is to perform the following tasks:

- All Space Station systems management
- Formatting and collection of all Space Station telemetry for downlink to the Mission Control Center
- Unpacking and distribution of all Space Station commands uplinked from the Mission Control Center and generated by the flight crews aboard the Space Station from laptops
- Support of Payload Resource Management

In addition to the flight crews on the vehicle itself, the Space Station program contains support from the following control centers:

- Mission Control Center (JSC, Houston)
- International partner control centers (Japan, Germany, Canada, and Russia)
- Payload Operations and Integration Center (Huntsville, Alabama)

SSTF Components

The SSTF is a multinational simulator project being developed at JSC. The primary functions of the SSTF are to train the international flight and ground crews of the international Space Station and to verify the operational and mission procedures during in-orbit exercises and assembly flights. This can be provided in a time and cost efficient manner by integrating existing facilities into the virtual simulation. The SSTF provides a realistic mission rehearsal environment by taking advantage of the following existing NASA engineering capabilities:

- Shuttle Mission Simulator
- Weightless Environment Training Facility
- Payload Operations and Integration Center
- International Partner simulators:
  - •• Japanese Experiment Module
  - •• European Attached Pressurized Module
  - •• European Automated Transfer Vehicle
  - •• Soyuz
- Huntsville Operations Support Center
- Payload development center developed simulators
- Mission Control Center - Houston
- International Partner Control Centers

The SSTF virtual simulation achieves a dynamic configuration by allowing any of its assets to be added or dropped "on-the-fly." For example, if a virtual simulation is occurring with the JSC and Russian elements and crews from the Shuttle wish to join the simulation session, then the Shuttle Mission Simulator is simply added while the simulation is in the mode "FREEZE." After the controlling simulation is content that the various facilities are synchronized, the simulation is then put back into "RUN" mode.

By utilizing this implementation, the SSTF has experienced payoff from virtual simulation concepts by adapting to a number of potentially devastating events:

- The SSTF architecture has survived intact despite two major Space Station redefinitions.
- The payload simulation capability have been deleted from the SSTF, and then added back without perturbation to the overall system design.
- The Space Station assembly sequence introduces an unprecedented change factor into the SSTF configuration and functionality.
- The rapid change in computation technology over the Space Station's long life cycle has caused the SSTF to remain flexible in order to utilize the latest hardware and software products.

Software In The Loop

One of the ways the SSTF program has been able to survive the many changes which have occurred in the Space Station program is by adopting an aggressive reuse strategy. Throughout the Space Station program many engineering simulations are being produced by the vendors, manufacturers, and various government organizations. By reusing these pieces, the virtual simulation has gained high fidelity components with little or no software development by the SSTF program. The following are some of the reuse components which have saved software development effort.

Space Station Remote Manipulator System (SSRMS): The SSRMS is a simulation of a multi-segment articulated end-effector (i.e. robotic arm). This model was verified and validated by the JSC Engineering Directorate and contains complex dynamics and contact detection algorithms for the Space Station's robotic arm. Although developed outside of the SSTF project in C and FORTRAN, the model was wrapped with a real-time partition to handle simulation events and interfaces to other simulation models (e.g. power, thermal, C&DH, visual system etc...).

Electrical Power System Simulation (EPSIM): The EPSIM is a simulation of the Space Station electrical power system which is used for storing power from the Space Station's solar arrays and distributing that power to the many black boxes onboard the station. This simulation was developed by Rocketdyne as an engineering development tool to validate the performance of the electrical power system. Initially is was integrated into the SSTF as a separate external simulator. Currently, Rocketdyne is porting the simulation into the SSTF software architecture.

Command and Data Handling System Flight Software: Flight software development for the Space Station program is a huge undertaking due to the numerous computers in the C&DH system which do everything on the Space Station from communications and tracking to guidance navigation and control. To accurately simulate this system would drive the simulator cost up way over the allocated budget and probably never match the fidelity needed. As a result, the SSTF will use the real-world flight software being developed by Boeing (the prime contractor) and the various product groups (included the international partners). Also, the flight software will be run on Flight Equivalent Units (FEUs) of the real-world Space Station computers. To further reduce cost, an emulation of the 386 based flight computers is being developed. With real-world flight software, the fidelity of the simulation is guaranteed, and verification of the flight software can occur with a full system ETE test.

Simulation In The Loop

In addition to reusing existing software products, the SSTF has experienced integration of other simulators. These simulators have been developed by international partners using the SVM infrastructure or have existed for many years and were simply folded into the virtual simulation with interface agents.

At the geographic level, the SSTF has performed an early test with the Russian Space Agency. This test exposed the need to address traditional distributed simulation problems such as database correlation, time homogeneity, and transport lag. As a result, Distributed Interactive Simulation (DIS) techniques will be used to integrate the SSTF with remote simulations of other space vehicles to simplify interfaces. Also, the effects of loop lag transport will be minimized by using established algorithms (e.g. Clohessy-Wiltshire, Bang-Bang Attitude Controller Modeling).

At the local level, simulator integration is being used to expand the capabilities of the SSTF by supplying "plug'n'play" virtual simulation assets. In addition to the previously mentioned EPSIM, the following are some of the locally integrated simulation assets.

Shuttle Mission Simulator (SMS): The SMS is an old technology, high fidelity shuttle trainer. It has been used for years to train shuttle crews for mission operations. Based on a mainframe architecture, the majority of the code is written Concurrent Assembly Language (CAL) and FORTRAN. Via interface agents, the SSTF is using DIS techniques to accomplish integration, thus it is transparent to the SSTF whether the SMS is locally or geographically distributed. For Space Station operations, the SMS will be added to the SSTF as an asset to support simulations for Space Station/shuttle docking, docked operations, and material transfer.

Japanese Experiment Module (JEM) and European Attached Pressurized Module (APM): The JEM and APM will be provided by the National Space Development Agency of Japan (NASDA) and the European Space Agency (ESA) respectively. These simulators represent those modules of the Space Station which will be developed by the international partners. Integration with the SSTF will be performed using interface agent techniques.

Space Station payload simulators: Developed by several agencies throughout the United States, the payload simulators and packaged in Space Station Racks and will be installed into the SSTF Lab Module. These simulators communicate with the other models of the SSTF via Ethernet and MIL-STD-1553B by using a combination of standard interface definitions and interface agent techniques.

It is important to note that the SSTF may be run with or without any of the previously mentioned simulators. Capability is added or dropped depending on the needs of any given simulation session.

Hardware In The Loop

Sometimes, hardware will be used to fulfill the simulation requirements when a major piece of the simulation has not been developed by an outside organization, or software modeling is impractical. In this case, real-world or flight equivalent hardware is stimulated (software models driving hardware interfaces) or emulated (a "homemade" version of a real-world black box with simulation unique artifacts added). The SSTF has demonstrated integration with real Space Station vehicle hardware, either by stimulation or emulation with the following:

- Space Station On-Board Computers
- Control Centers
- Robotics Workstation
- Autonomous Vision Unit
- Flight Type Payloads

By including hardware in the loop (especially real-world flight equivalent units), the virtual simulation can evolve as a powerful ETE verification and test simulator. During the early stages of a project, the simulation will mostly contain software models. As the real-world hardware is developed, the software models can be "unplugged" and the hardware can be "plugged in" and stimulated by the rest of the models to test its functionality.

Conclusion

A virtual simulation architecture using Object-Oriented techniques, provides flexibility to assemble large scale ETE simulations. This is achieved by developing a homogenous infrastucture which allows integration of heterogeneous assets into the virtual simulation. When implemented effectively, a virtual simulator can reuse existing assets for greater fidelity and capability at a significant cost and time savings. As a result, a virtual simulator can be used for training, testing, and validation of, complex, distributed, real-time systems containing real-world hardware and software in the loop.

References

(1) Booch, G., *Object-Oriented Design*, Redwood City, CA: Benjamin-Cummings, 1991.

(2) Coad, P., and Yourdon, E., *Object-Oriented Analysis*, Second Edition, Englewood Cliffs, NJ: Yourdon Press, 1991.

(3) Rumbaugh, J., *Object-Oriented Modeling and Design*, Englewood Cliffs, NJ: Prentice-Hall, 1991.

(4) Clohessy, W. H., and Wiltshire, R.S., Terminal Guidance System for Satellite Rendezvous, Journal of Aerospace Sciences, Vol. 27, pp.653-658, Washington, D.C.: Institute of Aerospace Sciences, 1960.

(5) Smith, H., Sellke, J., Policella, J., and Stumpf, J., Multi-National Training in a Networked Simulation for the Space Station Training Facility, Proceedings of the 1996 Image Conference, Tempe, AZ: The Image Society, Inc., 1996.

A SIMULATION TRAINING FACILITY FOR SHUTTLE/MIR DOCKING

John M. Teel
Lockheed Martin Engineering & Science Services
Houston, Texas

Elizabeth M. Bains
NASA - Johnson Space Center
Houston, Texas

Y. M. Kuo
NASA - Johnson Space Center
Houston, Texas

T. K. Ghosh
Lockheed Martin Engineering & Science Services
Houston, Texas

Abstract

A real-time pilot-in-the-loop simulation was developed in the Systems Engineering Simulator (SES) to allow studies of docking the United States space shuttle orbiter to the Russian space station Mir. The SES facility, located at the National Aeronautics and Space Administration (NASA) Johnson Space Center (JSC), is being used for engineering studies and crew training for the US flights to Mir.

The SES simulation facility has supported, among others, studies of rendezvous of the orbiter with other orbiting vehicles and close approaches to those vehicles (proximity operations) since the early 1980's. For the Mir studies a critical part of the simulation was modeling the dynamics of the contact between docking mechanisms of the orbiter and the Mir in a way that was suitable for real-time simulation. The simulated dynamic vehicles in this simulation consist of the orbiter, Mir, and an active docking ring on the orbiter docking system. The Mir docking ring is rigidly attached to Mir. During the docking, contact is modeled between the petals and rings of the orbiter and Mir docking mechanisms.

In this paper, the software design and the technique of mathematical modeling of the contact dynamics that was used in this simulation are given. The validity of the simulation response in comparison with a high fidelity structural model is also discussed. Finally, a video showing the final phases of an SES simulation of an orbiter approach to Mir and the dynamics of the docking contact will be shown.

I. Introduction

Docking between the United States space shuttle orbiter and the Russian space station Mir is accomplished using a Russian developed docking system. The docking system currently being used is derived from the docking system used for the Apollo - Soyuz dockings in the 1970's and is more fully described in section II. The Russians have used this docking system to dock Soyuz and Progress vehicles to the Mir space station. Since the latches that provide the initial capture require a significant force to engage, the Soyuz (or Progress) approaches the Mir docking port with a relative speed high enough to provide the energy required to engage the latches. Each of these vehicles has thrusters in-line with its docking port so that a velocity along the axis of the docking port is easy to control.

For the U. S. orbiter, however, there are no jets in-line with the docking system, and all jets are canted with respect to the docking port axis. A command to thrust along the docking port axis produces jet firings that couple in translation and rotation. The standard practice when the orbiter captures a free-flying payload is to approach at low speed and come to a complete stop relative to it while the payload is captured by the Canadian Remote Manipulator System. Therefore, the requirement for the orbiter to approach Mir with sufficient relative motion to engage the docking system latches caused initial concern that cross coupling from the orbiter jets would prevent capture.

A simulation of orbiter approach to and docking with Mir was developed in the Systems Engineering Simulator (SES) at the NASA Johnson Space Center. The SES had previously been used extensively for developmental testing and operations training for orbiter approach and capture of free-flying payloads. In these studies the pilot positions the orbiter relative to the payload without making contact. For the Mir studies, however, the docking ports were likely to make contact before the latches were in position to engage. It was, therefore, necessary to add to the SES a dynamic model of contact between the docking ports for these studies.

Section II describes the Russian docking system and section III describes the SES. Sections IV to VI describe the contact dynamics model, its design for integration into the SES, and verification. Section VII describes the use of this model in the SES.

II. Docking System Description

The docking system consists of a passive docking ring assembly rigidly attached to Mir and an active docking ring mounted on the orbiter docking system in the orbiter payload bay. Figure 1 is a picture of the orbiter docking system in its extended position. This picture was taken by the Space Transportation System (STS)-71 crew. The orbiter docking system contains gears, springs, clutches, and dampers that allow the active docking ring to move during docking and absorb energy due to relative motion between Mir and the orbiter. Each docking ring consists of a ring with three evenly spaced petals. When the rings are mated, each petal fits into the space between two mating ring petals.

During docking, contact between the docking ring petals assists in aligning the docking rings. Three spring loaded latches on the active docking ring latch the rings together when the rings are properly mated. Docking system sensors drive indicators of initial contact and successful latching in the aft cockpit. After latching, dampers between the active docking ring and the orbiter automatically transition to a high damping mode to reduce the relative motion between the Mir and orbiter. Finally, the orbiter docking ring is retracted to form an air tight seal between the two vehicles.

If an emergency separation from Mir is necessary, the crew has the capability to issue a command to open the docking ring latches to allow separation.

III. SES Facility

The SES is a real-time human-and/or hardware-in-the-loop simulator that supports developmental testing and operations for Space Shuttle, Space Station, and advanced programs. Developmental testing typically includes evaluation of areas where human/machine interaction is a significant consideration. Operations support includes engineering evaluation, mission design and training, and real-time mission support when required.

The SES consists of two simulators: an ascent /entry simulator with a Shuttle forward cockpit and an on-orbit simulator with four cockpits/workstations that can be used independently or in an integrated simulation.

Response to cockpit or workstation inputs by an operator is driven by high fidelity dynamics of the appropriate systems. Out the window views and closed circuit television views, generated by an electronic scene generator, give the operator visual cues that would be available in the environment being simulated.

The on-orbit SES is easily reconfigurable to allows studies including multiple vehicles. Available vehicles and systems include the orbiter, Mir, one or more shuttle payloads (Wakeshield, SPARTAN, SPAS, etc.), the international space station, remote manipulator systems for either shuttle or space station, the tethered satellite dynamics, etc. Simulation of a system includes its control system, as appropriate, dynamics of the physical system, and significant effects of the on-orbit environment.

The orbiter - Mir docking studies used the on-orbit simulation and were conducted in an orbiter aft flight deck mockup. The simulation includes gravitational effects on each orbiting vehicle (including gravity gradient), aerodynamic effects, and forces due to impingement of gases from the orbiter Reaction Control System (RCS) jets on orbiter surfaces (self plume) and on Mir. Special modeling for the docking studies, in addition to the docking contact model described in section IV, included the docking port camera view and docking cues and switches that are available on-orbit. The docking port camera is mounted on the orbiter docking system and is aligned with a target on the Mir docking port when the vehicles are perfectly aligned for docking. Docking cues include an initial contact indicator and a capture indicator. An open latches switch is available for simulation of emergency and nominal release of the orbiter from Mir. An audible cue for contact between the docking rings is available.

IV. Docking Contact Dynamics Model

The SES simulation model of the docking system with contact dynamics is designed to interface with other SES models to take advantage of the existing capabilities of the SES. The math modeling accounts for the requirement to perform calculations in real-time.[1]

The orbiter, Mir and the active docking ring are modeled as three free flying rigid bodies. The passive docking ring is modeled as a rigidly attached part of the Mir body. Dynamic motion of each body is simulated by numerical integration of six-degree-of-freedom equations of motion (EOM). Simulated external forces and moments along with gravitational force acting on the bodies are accounted for in the EOM model to propagate the states of the bodies to the next simulated time step. A docking contact dynamics model calculates all forces and moments associated with the docking system on the three bodies. These forces are added to other external forces acting on the bodies and used by the EOM model. Forces and moments exerted by the orbiter docking system on the active docking ring and the orbiter are modeled with an uncoupled spring and damper for each of the six relative degrees of freedom between the orbiter and active docking ring. A friction clutch is modeled along with the spring and damper for the axial degree of freedom. Linear and angular offsets of the active docking ring from an equilibrium position are taken as spring deflections. Restoring forces and moments acting on the active docking ring are computed from spring and damper characteristics. The spring characteristics include hysteresis effects of the docking system.

For modeling contact between the active and passive docking rings, each docking ring petal is represented by two straight line edges and a centerline. The rings are represented by circles. Contact forces are calculated from the penetration distance and penetration rate of one surface into the other using a contact spring and damper model.

A spring model is used to simulate docking ring latch forces that must be overcome to successfully latch the passive and active docking rings together. Forces and corresponding moments are calculated based on the distance each active docking ring latch is depressed by the passive docking ring latch surface.

Criteria for latching the docking rings together are based on the height of three passive docking ring latch points above the base plane of the active docking ring. Partial latching is not modeled. When latching criteria are met, the active docking ring moves into perfect alignment with the passive docking ring. The active docking ring is then treated as rigidly attached to the Mir body. Forces and moments from the orbiter docking system are applied to the Mir body when the docking rings are latched and no contact between the docking rings is modeled. Emergency release is simulated by detaching the active docking ring from the Mir body and applying contact and orbiter docking system forces and moments to the active docking ring again.

Nominal release is handled as a special case. Three loaded springs on the orbiter docking system are simulated to provide push-off forces and corresponding moments to separate the Mir and orbiter.

Retraction of the orbiter docking ring is not modeled in the SES since no crew input is required during retraction.

V. Software Design

The docking contact dynamics model is implemented in FORTRAN. The SES simulation is set up to use a 40 ms time frame for scheduling the execution of math model software. However, the contact dynamics model is unstable before latching if executed every 40 ms (25 Hz). The spring characteristics of the simulated orbiter docking system are designed to produce forces that act between the mass of Mir and the mass of the . The springs of the docking mechanism are designed for the heavy masses of the Mir and the orbiter. They are very stiff for the small mass of the active docking ring alone. Integration of the EOM for the active docking ring before latching at the 25 Hz integration frequency, therefore, leads to numerical instability. To avoid this instability, EOM calculations for the orbiter, Mir, and the active docking ring are executed 7 times in a loop within the contact dynamics model every 40 ms to achieve an effective frequency of 175 Hz. Once latching occurs, the spring forces act directly on the masses of Mir and the orbiter, so the 25 Hz frequency is sufficient and the inner loop is reduced to a single pass.

VI. Verification

The SES docking contact dynamics model was verified by comparing simulation results with a non-real-time, high fidelity simulation developed by NASA Structures and Mechanics Division (SMD) at the Johnson Space Center. Simulation runs were started just prior to contact and continued for a few seconds after latching. Figure 2 shows reference frames used to define the simulation variables. An orbiter docking port frame (ODP) is fixed to the orbiter at the active docking interface and a Mir docking port frame (MDP) is fixed to Mir at the passive docking interface. Figures 3 through 9 are cross plots of various variables comparing the SES simulation to the SMD simulation. For this particular test case, the active docking ring is initially misaligned from the passive docking ring by two inches in the orbiter aft direction. The initial closing velocity is 0.1 feet per second purely along the ODP X axis. Figures 3 and 4 show respectively the ODP frame X axis relative position and velocity between the ODP and MDP origins. Figure 5 shows the pitch angle between the ODP and MDP frames. Figure 6 shows the total docking force on the orbiter along the X ODP direction. Figures 7 through 9 show the total docking contact force on Mir before latching.

VII. Use of the Simulation

The SES orbiter - Mir docking simulation has been used extensively for engineering studies and crew training. Initial studies focused on the ability of orbiter pilots to fly a high speed approach, as the Russians did, and still position the orbiter accurately enough to achieve a good docking. It was found to be feasible, but orbiter jet cross coupling required many corrective firings close to Mir and pilots were generally uncomfortable with the approach. The close in jet firings also increased the contamination and loads on Mir from RCS jet plume impingement. Because of these disadvantages of a high speed approach, a special mode in the on-orbit Digital Auto Pilot was developed to allow a slow approach to Mir.

The new mode, Post Contact Thrusting (PCT), fires a sequence of jets that gives a thrust along the docking port axis that is high enough to engage the latches. After the orbiter has approached slowly and made contact between the two docking ports, PCT is initiated to engaged the latches.

Other studies have included comparison of approach along the velocity vector (v-bar) of the Mir orbital velocity or along the radius vector (r-bar) from the Mir to the center of the earth. The v-bar approach is more stable since the orbiter and Mir are in the same orbit if the orbiter holds position relative to Mir. However, during an approach, jets must be fired toward Mir (causing contamination and loads) if it is necessary to slow or stop the approach. The r-bar approach from a position below Mir has the advantage of natural braking as the orbiter moves to a higher orbit to dock with Mir. If the orbiter stops during the approach, it is necessary to fire jets frequently to maintain position relative to Mir, but the firings are not toward Mir. The r-bar approach from below Mir has, therefore, been used for all Mir dockings from STS-71 through STS-79 and is currently planned for all future Mir dockings.

Other engineering studies have included an inertial approach which was requested by the Russians to allow Mir solar arrays to remain pointing at the sun. This required very high fuel usage by the orbiter and was not adopted.

The SES orbiter-Mir docking simulation is used by each crew flying an approach to Mir. The simulation can begin with a rendezvous from between 9000 feet to 13,000 feet or can begin as a proximity operations approach from about 400 feet. The crew initially aligns the orbiter using the Crew Optical Alignment Sight (COAS) in the starboard overhead window. Range and range rate cues are obtained from either an orbiter mounted Ku-band radar, a laser mounted in the orbiter bay, or a hand-held laser.

When the orbiter is close enough, the pilot uses the view from the docking port camera to assist in alignment of the docking rings. After initial contact is annunciated by the contact light in the cockpit, the pilot initiates the PCT firings to drive the docking rings together and achieve latching. An approach that doesn't allow proper alignment of the docking ring will result in a failed capture. The crew trainer can also choose to disable capture for a particular run to train the crew in recovery techniques.

Crew training in SES with the contact dynamics models is an integral part of the training that gives crew good confidence for on-orbit operations. In addition, this capability is available or can be expanded to address other engineering and operational issues as they arise.

Acknowledgments

The authors acknowledge the following for supporting this effort: John D. Schliesing, NASA Johnson Space Center, and Julia Chu, Lockheed Martin Engineering & Science Services.

References

1. Simulation Systems Branch, Systems Engineering Division, Johnson Space Center, "System Engineering Simulator On-Orbit Element Simulation Definition Document," LEMSCO-2411, Rev. A, January, 1991.

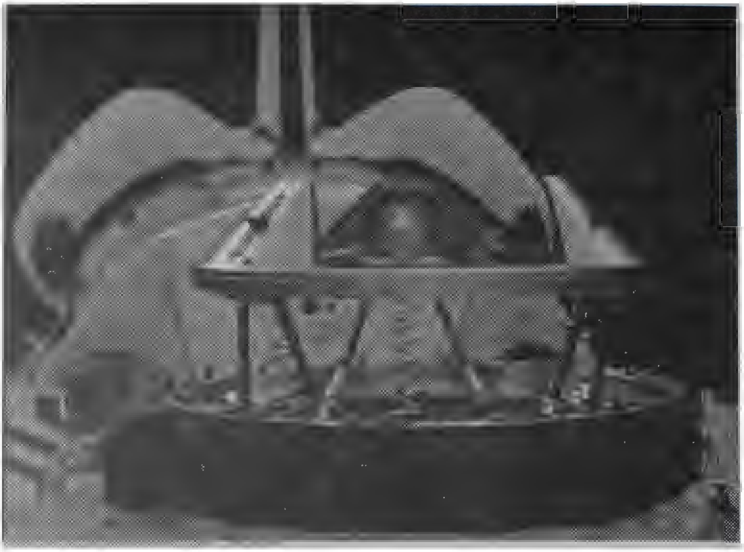


Figure 1 - STS-71 orbiter docking system.

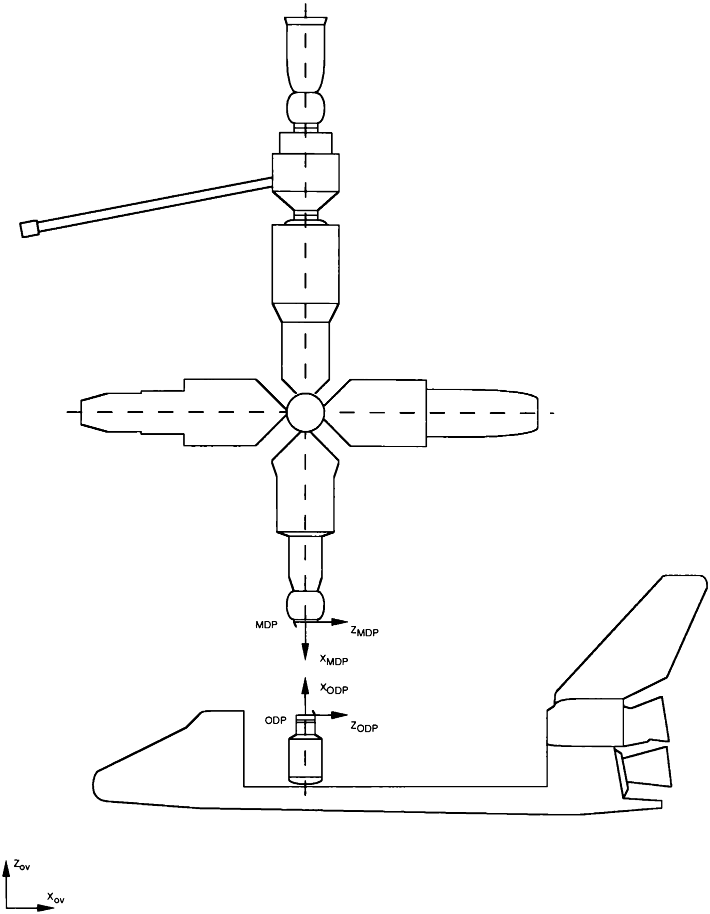


Figure 2 - STS-71 Orbiter and Mir with docking port reference frames

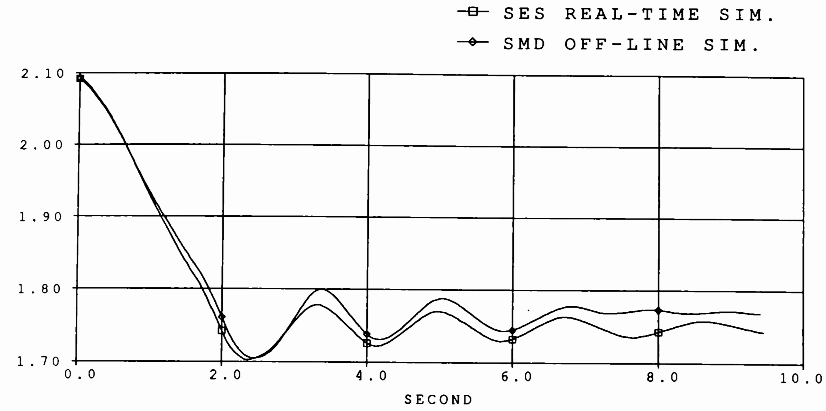


Figure 3 - The MDP frame origin relative to the ODP frame in X-axis (ft).

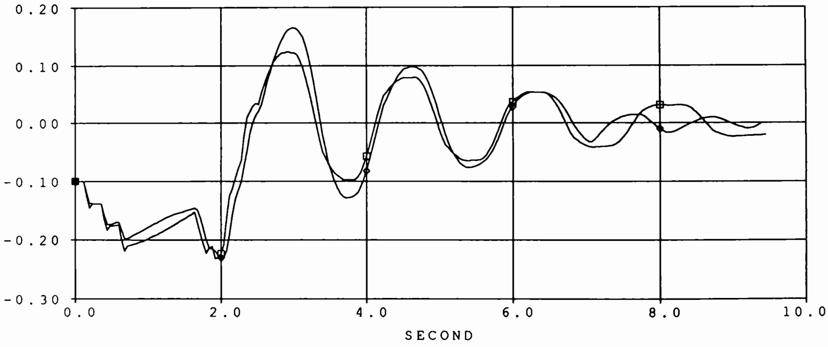


Figure 4 - The MDP frame velocity relative to the ODP frame in X-axis (ft/sec).

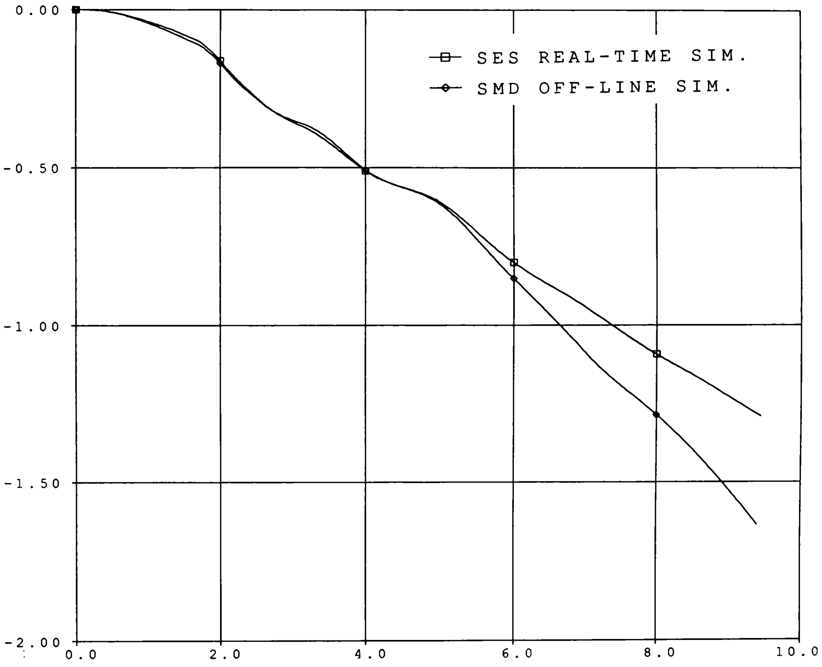


Figure 5 - The MDP frame pitch attitude relative to the ODP frame (deg).

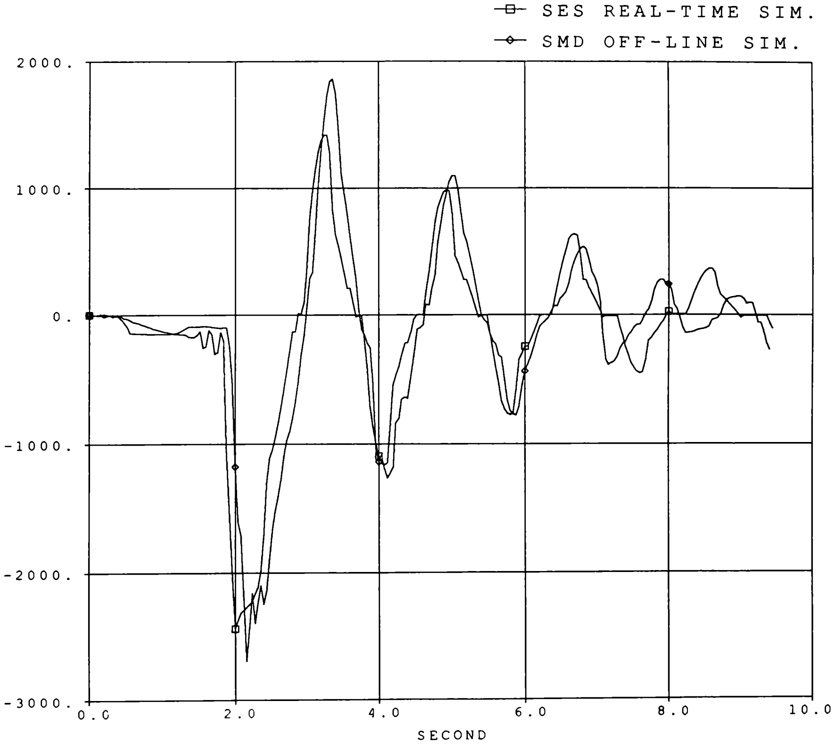


Figure 6 - The X-axis docking force on the orbiter in the ODP frame (lbs).

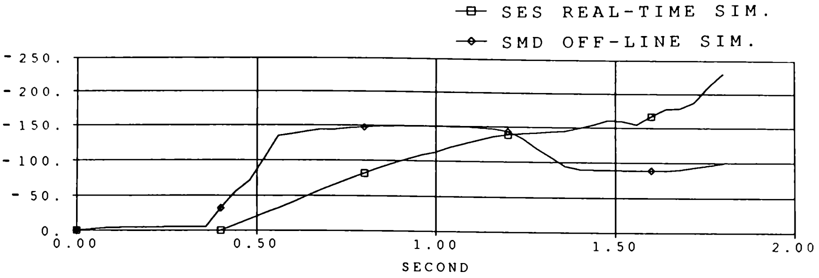


Figure 7 - X-axis docking force on Mir in the MDP frame (lbs).

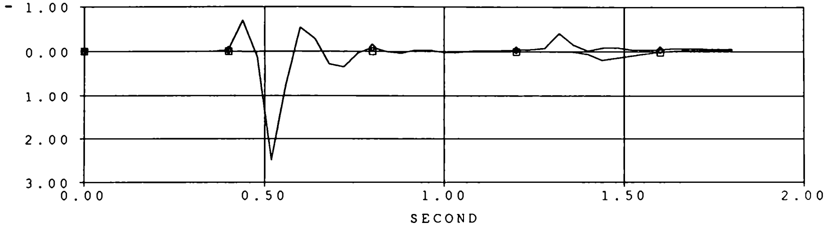


Figure 8 - Y-axis docking force on Mir in the MDP frame (lbs).

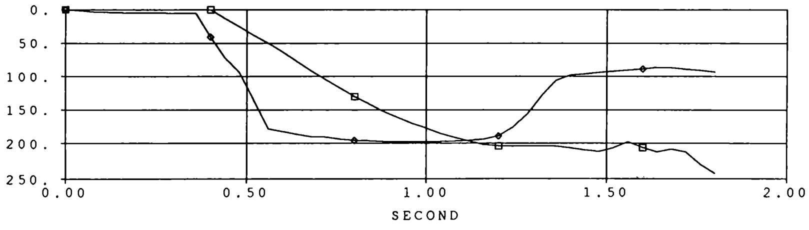


Figure 9 - Z-axis docking force on Mir in the MDP frame (lbs).

AIAA-96-3498-CP

AN INTEGRATED EVA/RMS VIRTUAL REALITY SIMULATION, INCLUDING FORCE FEEDBACK, FOR ASTRONAUT TRAINING

David J. Homan and Charles J. Gott
Automation, Robotics, and Simulation Division
NASA Johnson Space Center, Mail Stop ER
Houston, Texas 77058

Abstract

The application of virtual reality (VR) technology to ground-based training of astronauts in preparation for extravehicular activities (EVA) on space shuttle missions was initially evaluated in conjunction with the first Hubble Space Telescope (HST) repair mission in 1993. This initial proof-of-concept application was used by the Remote Manipulator System (RMS) operator and an EVA crew member positioned on the end of the RMS to establish and validate a command protocol for directing movements of the RMS during integrated operations. The intravehicular (IV) crew members also participated in some simulation sessions to help familiarize themselves with the EVA scenarios. A second application was developed to support training in the use of the Simplified Aid For EVA Rescue (SAFER) unit, flight tested on STS-64 in the fall of 1994. The helmet-mounted display (HMD) was integrated with the SAFER avionics, hand controller hardware, and flight software to create a simulation which provided a 3-D graphic representation of the orbiter payload bay and RMS configurations as seen from the vantage point of the SAFER crew member. Again this application afforded a capability to train in an integrated environment not available in other ground-based simulators. Another important aspect of EVA is the physical handling of large objects under weightless conditions. While exploring the effectiveness of the virtual reality simulation with the HST crew members, a concept to realistically simulate the inertia of payloads being manipulated in the VR environment was proposed. This capability was developed and evaluated in the JSC Integrated EVA/RMS Virtual Reality Simulation Facility using a force feedback device to simulate the zero-g mass characteristics of large (>500 lb.) orbit replaceable units (ORU). A tendon-driven robotic device provides the kinesthetic sensation of the mass and inertia characteristics of the object being handled, while the HMD provides the VR subject with a visual representation of that object and its surroundings. Integration of a second mass simulator into the system allows 2 EVA/VR crew members to cooperatively manipulate the same large mass. Results from these initial demonstrations have shown great potential for VR technology to support ground-based preparation for on-orbit integrated EVA/RMS operations as well as EVA free-flyer piloting. The simulation has been used to support integrated EVA/RMS training and task development for all shuttle flights involving extravehicular activities.

Introduction

It is important to emphasize up front that VR has not reached, and will not reach in the foreseeable future, a level of maturity or sophistication adequate to allow it to replace any of the current EVA or RMS simulation/training facilities. What it does offer, however, is an arena in which to integrate the specific training and simulation capabilities of these facilities in a more coherent manner than has been possible in the past.

For the first HST repair mission, integrated operations between the Remote Manipulator System (RMS) operator and an EVA crew member positioned on the manipulator foot restraint (MFR) at the end of the RMS were practiced in the VR simulation (Fig. 1).

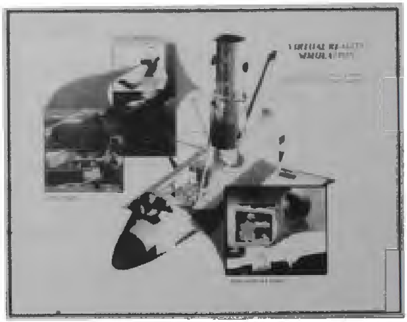


Figure 1. Integrated EVA/RMS VR simulation for HST.

While the RMS operator controlled the position and attitude of the MFR around the HST using the desktop RMS part-task training simulation, the EVA crew member wore a head-mounted display (HMD) that provided a computer generated 3-D graphic representation of the actual on-orbit configuration as would be viewed by the EVA crew member at the end of the RMS. The EVA crew member's head motion was sensed, as were the motions of the hands, by electromagnetic tracking devices attached to the HMD and to each of the data gloves worn by the VR subject, thus allowing the computer graphic scenes to be realistically portrayed for both the RMS operator and the EVA crew member. The VR simulation was used to enhance the training provided by the neutral buoyancy facilities located at both the Johnson Space Center (JSC) (Fig. 2) and the Marshall Space Flight Center (MSFC) (Fig 3.), as well as the Manipulator Development Facility (MDF) (Fig. 4) and the Shuttle Mission Simulator (SMS).

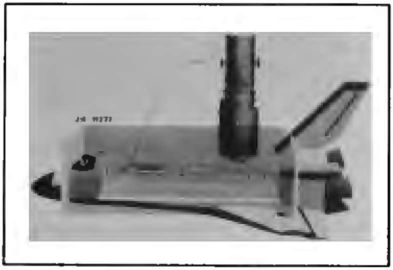


Figure 2. Weightless Environment Training Facility (WETF).

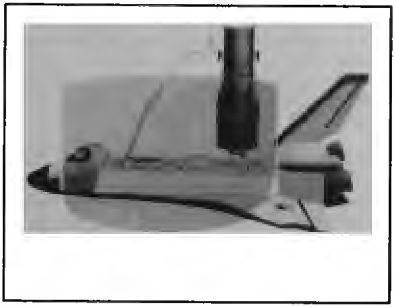


Figure 3. Neutral Buoyancy Simulator (NBS).

Figure 4. Manipulator Development Facility (MDF).

These ground-based simulation facilities offer high fidelity part-task training in EVA and RMS operations, respectively, but are restricted by safety, astronaut comfort, RMS lift capability, and volumetric shortcomings from adequately representing integrated EVA/RMS operations over the entire range of on-orbit geometric configurations.

In a joint effort to understand and quantify the capabilities and limitations of VR technology to support meaningful and productive applications, the Astronaut Office, EVA/RMS training personnel, and engineers from the Automation, Robotics, and Simulation Division (AR&SD) are continuing to develop, evaluate and enhance applications in support of on-going Space Shuttle missions to prepare for future assembly, maintenance and operations of the International Space Station.

System Capabilities

The basic capabilities available in the Integrated EVA/RMS VR Simulation facility at JSC include:

RMS/IV Work Station

This workstation includes a functionally correct dynamic simulation of the Shuttle Remote Manipulator System with translational and rotational hand controllers, graphical representation of the A8 control panel, access to all pertinent shuttle general purpose computer (GPC) Spec pages, and an active input key pad (Fig. 5). The simulation also provides access to all closed circuit television (CCTV) views available from all payload bay and RMS cameras and out-the-window views.

Figure 5. RMS/IV Workstation.

EVA Positions

Each of the two positions (Fig. 6) consists of a low resolution, wide field-of-view helmet mounted display (HMD) built by Virtual Research or a high resolution, wide field-of-view Datavisor 80 built by n-Vision. The views in each HMD are tuned to match the eye characteristics of the wearer to reduce eye strain and fatigue. The motions of each subject's head, body, both hands, and one object being manipulated are tracked with five Ascension electromagnetic trackers. Each position has a pair of data gloves to allow each crew member to interact with active objects, such as handrails or other handling devices.

Figure 6. EVA VR Simulation.

SAFER Simulation

The simulation includes crew member mass characteristics, orbital dynamics, jet firings, fuel and battery power usage. The simulation is driven by actual flight avionics boards and a flight-like hand controller (Fig. 7). Following the engineering flight test of the first SAFER unit on STS-64, the crew reported that their ability to fine tune the maneuvers flown during the EVA evaluation of the system was attributable to the virtual reality training they received prior to their flight.

Figure 7. STS-64 crew member in VR SAFER Simulation.

Mass Handling

One of the most important aspects of EVA is the physical handling of objects such as ORUs. While exploring the effectiveness of AR&SD's VR simulation in sessions with the Hubble Space Telescope (HST) crew, the AR&SD/ McDonnell Douglas Aerospace (MDA) team developed a concept to simulate the inertia of payloads being manipulated in the VR environment. The concept employed Charlotte, an intravehicular robot (IVR) developed by MDA for use inside the SpaceHab, SpaceLab, and Space Station modules in a zero-g environment. Because it is designed to work along side a shuttle or station crew, it is inherently safe, light weight, low power, easy to set up, quiet, and reliable.

Charlotte was modified (Fig. 8) to incorporate a force/ torque sensor, EVA hand holds, equations of motion, envelope checks, and safety checks but required little changes for safety concerns because of parallel efforts to certify the design to NASA-STD-3000 for flight operations with the crew. This new force feedback device was named Kinesthetic Application of Mechanical Force Reflection (KAMFR) and incorporated as part of the VR training simulation.

Figure 8. Modified Charlotte Robot

The integration of KAMFR with the existing VR system allows data to be transferred from a dynamic simulation to KAMFR and from KAMFR to the VR simulation. Using this architecture the VR simulation environment fully supports combined RMS/EVA operations with a workstation for the RMS operator and a virtual environment for the EVA crew member. This mass handling simulation (Fig. 9) has, to the satisfaction of the EVA crew members involved, successfully replicated the zero-g mass characteristics of the 2700 lb. Spartan payload manually handled during the EVA Development Flight Test-01 (EDFT-01) which was performed on STS-63 (Feb. 1995). A second robot is currently being integrated into the simulation to allow two crew members to manipulate the same large mass at the same time

Figure 9. STS-63 crew members evaluating KAMFR.

Data Recording and Playback

The system has the capability to make real-time audio and video recordings of all positions for documentation purposes and motion data recordings of all positions for post-simulation playback and evaluation. Recording of all motion data also allows interactive playback with the simulation so that crew members can view previous simulations from within the environment.

System Configurations

The use of VR as an EVA training device has evolved over time to match the advances in hardware technology and the varied training requirements as applied to each flight. Following are synopses of each shuttle flight supported by the Integrated EVA/RMS Virtual Reality Simulation Facility:

STS-61 HST Repair (Fig. 10)

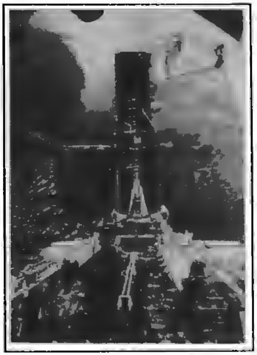


Figure 10. EVA crew member on RMS.

- December 1993
- One EVA crew member and one RMS operator.
- One Helmet Mounted Display (HMD), two DataGloves, handles representing HSP/COSTAR handholds and jettison tool.
- 8 sessions (~ 16 hrs)
- Developed command protocol between EVA and RMS operator for maneuvering the RMS while crew member was attached to the end.
- Removed HSP from HST, transferred it to temporary holding device, removed COSTAR from ORU carrier and positioned it in HST, moved HSP from temporary storage location to ORU carrier.
- Reworked RMS positions for magnetometer replacement at top of HST.

STS-60 WSF contingency retrieval (Fig. 11)

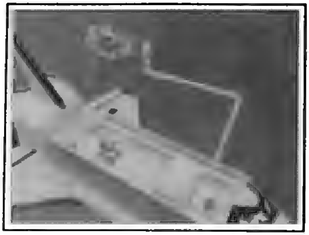


Figure 11. WSF contingency retrieval by EVA

- February 1994
- One EVA crew member and one RMS operator.
- One Helmet Mounted Display (HMD), two DataGloves.
- 1 session (1.5 hrs)
- Evaluated manual retrieval of tumbling WSF by EVA crew member on the end of the RMS.
- WSF motion driven by dynamic simulation integrated into the VR simulation.

STS-64 SAFER demonstration flight (Fig. 12)

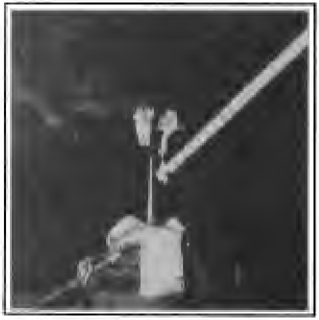


Figure 12. SAFER crew member and RMS.

- September 1994
- One EVA crew member, one RMS operator, IV crew members.
- One Helmet Mounted Display (HMD), SAFER hand controller.
- Approx. 60 hrs SAFER training for 2 EVA crew members with approx. 24 hrs of IV participation.
- Simulated all SAFER DTO scenarios.
- Integrated RMS operations as they pertained to SAFER DTO (i.e. rescue demonstration, EVA crew member on end of RMS caused SAFER crew member to tumble after which RMS operator moved the RMS away at a constant rate to simulate SAFER crew member inadvertent separation).
- Simulation used to baseline fuel consumption, timelines, handling characteristics, IV camera positioning, etc.

STS-63 EVA Development Flight Test (Fig. 13)

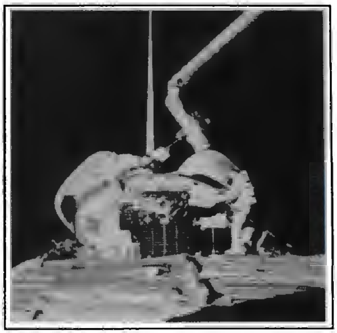


Figure 13. EVA crew members with Spartan.

- February 1995
- Two EVA crew members, one RMS operator, IV crew members.
- Two HMD's, two body sensors, four DataGloves, two payload handling devices.
- 5 sessions (7 hrs)
- Practiced integrated EVA/RMS operations in preparation for mass handling demonstration on orbit. Developed command protocol for handing Spartan payload from EVA crew member in fixed foot restraint to crew member on RMS. Evaluated RMS operations used in conjunction with mass handling demonstration. IV crew setup CCTV views and choreographed EVA checklist/timeline.
- Post-flight evaluation of force feedback device to simulate mass characteristics of Spartan payload.

STS-69 EVA Development Flight Test (Fig. 14)

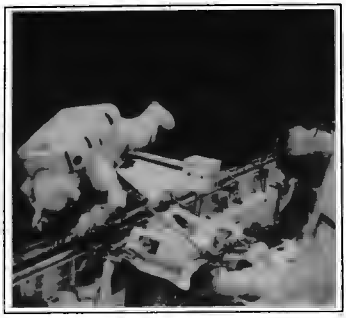


Figure 14. EVA in payload bay.

- September 1995
- Two EVA crew members, one RMS operator, IV crew members.
- Two HMD's, two body sensors, four DataGloves, one payload handling device.
- 7 sessions (18 hrs)
- Practiced integrated EVA/RMS operations in preparation for on-orbit demonstrations.
- Developed RMS operations to position EVA crew member for tasks in conjunction with WETF runs.
- Used hand-over-hand translation capability of simulation for EVA crew member positioning.

STS-75 Tethered Satellite System (Fig. 15)

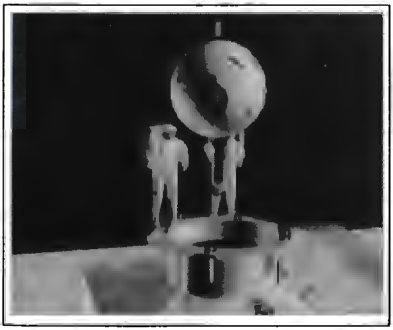


Figure 15. Contingency EVA planning.

- February 1996
- Two EVA crew members, IV crew member.
- Two HMD's, two body sensors, four DataGloves, two payload handling devices, one pair of EMU legs.
- 6 sessions (10.5 hrs)
- Evaluated crew positions for reach and visibility to support contingency operations using PFRs, PFR extensions, and one crew member holding the other crew member by the legs.
- Used force feedback system to simulate mass characteristics of TSS to evaluate handling of the TSS and the deployer mast.

STS-76 EVA Development Flight Test (Fig. 16)

Figure 16. EVA on MIR docking module.

- March 1996
- Two EVA crew members, one IV crew member.
- Two HMD's, two body sensors, four DataGloves, SAFER hand controller.
- 7 sessions (15.5 hrs)
- Crew familiarization of translation paths and MEEP deployment sequence.
- SAFER training.

STS-80 EVA Development Flight Test (Fig. 17).

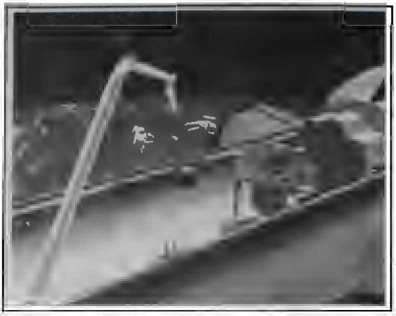


Figure 17. STS-80 payload bay configuration.

- November 1996
- Two EVA crew members, one RMS operator.
- Two HMD's, two body sensors, four DataGloves
- Two integrated KAMFRs.
- Integrated EVA/RMS operations development, WSF contingency retrieval.
- Mass handling evaluation with simulated station battery box (500 - 1000 lb.)
- Evaluation of multi-person mass handling.

STS-82 HST servicing mission (Fig. 18).

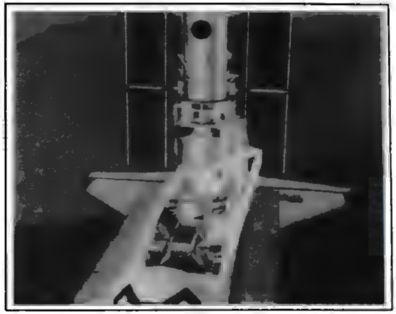


Figure 18. STS-82 payload bay configuration.

- February 1997
- Two EVA crew members, one RMS operator.
- Two HMD's, two body sensors, four DataGloves
- Two integrated KAMFR's.
- Integrated EVA/RMS operations development.
- Mass handling HST ORU's (500 - 1000 lb.).

System Architecture and Performance

The current system architecture is depicted in Figure 19. The processes are networked using a variety of communication protocols, such as ethernet, shared

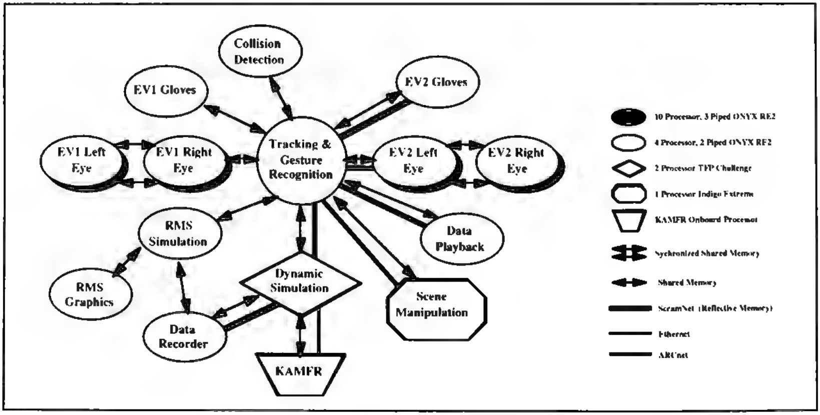


Figure 19. Current VR training system architecture.

memory, and reflective memory. Communication between each process is handled using the fastest protocol available. For example, two programs running on the same machine will use shared memory, where as two programs running on two different machines having both reflective memory(Scramnet) and ethernet will default to reflective memory protocol.

In addition to network flexibility, the communications subsystem maintains all changes made to the global environment. For example, changes made to the VR environment by one process are automatically reflected to all other processes. The system is also designed to handle the arbitrary termination or execution of any process in the distributed system. The state of the global environment is maintained and automatically restored as long as one process in the distributed system is left running. This allows processes to be brought on-line as needed, thereby reducing the overall load on the system.

The real strength of the communications subsystem is the straight forward and robust application programming interface (API) that allows full integration of a preexisting simulation into the VR environment in a matter of minutes instead of days.

The performance characteristics of the system are shown in Table 1. The performance numbers are based on Silicon Graphics Reality Engine2 graphic subsystems.

Table 1. System performance.

Triangular Polygons	Transformation Nodes	Refresh Rate
11,000	115	30 Hz. Constant
23,000	133	15 Hz. (Min.)
40,000	236	12 Hz. (Min.)
53,000	417	10 Hz. (Min.)

Conclusion

The use of virtual reality to support ground-based preparation for EVA assembly and maintenance of the International Space Station has been demonstrated to be a viable and productive application of the existing technology. In addition, the applications and system hardware/software architectures currently under development provide the cornerstone for a VR training/familiarization system on board International Space Station.

There are five major technical areas of importance to virtual reality simulation.

- Computer graphics or display rendering
- Helmet mounted display capability
- Tracking/sensor technology
- Kinesthetic display or force feedback capability
- System architecture and application development

The first three areas are being driven by powerful "outside" forces such as the entertainment industry and the home and business computer market—the requirements of which meet or exceed NASA's requirement to support virtual reality simulation capability development for the foreseeable future. The appropriate strategy becomes one of being prepared to use the technology when it becomes available as opposed to driving the technology to meet an application.

Kinesthetic display and force feedback, will be most effective and cost efficient if developed for specific applications or families of similar applications, in lieu of trying to develop a generic "one system does all" approach. Currently large-mass handling by two crew members is the focus of the effort in this area.

The final item, system architecture and application development, affords the largest return on NASA's investment in virtual reality. The virtual reality software architecture is being developed to support advancements in hardware technology as well as being tailored to accommodate the evolving requirements for integrated/interactive EVA simulation capability.

The development approach being taken to support today's EVA/RMS training requirements for virtual reality simulation lays the groundwork for understanding and applying the lessons learned to the areas of telepresence and teleoperation of robotic devices. These lessons learned include not only the technology but also the human interaction with that technology.

References

Homan, D., (1994). "Virtual Reality and the Hubble Space Telescope", AIAA Space Programs and Technologies Conference, Huntsville, AL, AIAA 94-4558.

Testa, B., (Winter 1994). "Virtually Walking in Space", Virtual Reality Special Report, pp: 67-74.

CREW ESCAPE SYSTEMS INTERACTIVE MULTIMEDIA COMPUTER BASED TRAINER

John J. Maca III
Ronald B. Lee
Simulator Operations and Technology Division
NASA Johnson Space Center
Houston, TX 77058

Abstract

The Crew Escape Systems Multimedia Computer Based Trainer (CBT) was developed to support astronaut Shannon Lucid on-board the Mir Space Station during the LDM-2 mission. The CBT will provide in-flight refresher training on the Crew Escape Systems procedures and equipment before she returns to Earth on the Shuttle. The CBT consists of two lessons covering the emergency egress procedures, crew worn equipment and related Shuttle equipment. The lesson content will change depending on whether it is taken as part of ground-based training or in-flight during a mission and the crew worn equipment lesson can be taken in Russian. Additional features include a 'Knowledge Challenge' and the ability to 'Review a Topic' versus taking the complete lesson. These lessons were developed with the goal of providing training "anytime-anywhere" and to show the potential for future in-flight training. The CBT was developed in-house by the authors using a rapid application development approach. This paper addresses the technical and project management aspects related to developing this CBT lesson for the Field Deployable Trainer Prototype project. It will discuss the development strategy, tools, processes, lessons learned and associated caveats. Included are the project development metrics and cost data in man-hours.

Introduction

There has been little in-flight astronaut training during a Shuttle mission. The longest Shuttle mission to date has been 16 days. With the joint cooperation between the US and Russia, astronauts will be staying on board the Russian Space Station Mir for several months. This will also be the case when the International Space Station becomes operational. The capability to conduct in-flight training during these long duration mission will be a necessity.

Training on the Crew Escape System is crucial for survival in the event the crew must rapidly egress the orbiter in an emergency situation. Although these procedures have rarely been used (bailout has never been performed) during a mission, the material covered in these lessons could mean the difference between life and death. While on-board Mir, subject data suggest the astronaut will experience cognitive degradation and refresher training on topics needed for return on the Shuttle is required. For example, Dr. Norm Thagard spent approximately 90 days on-board Mir during the LDM-1 (Long Duration Mission-1.) During his post-mission debrief, Dr. Thagard noted that it would have been beneficial to have some type of refresher training on the Shuttle Crew Escape System and the Launch and Entry Suit prior to his return flight on the Shuttle. As a result of his comments, this project was initiated to rapidly develop a CBT lesson for in-flight and ground-based training to support astronaut Shannon Lucid during the LDM-2 mission.

These CBT lessons were developed in-house by NASA personnel for the Field Deployable Trainer (FDT) project using a rapid development approach. The FDT is a research project to develop the methodologies and processes necessary to support on-orbit and portable remote training in support of human space flight. This was the first interactive multimedia computer based training lesson developed for pre-flight and in-flight use. There were two lessons developed in this effort; Crew Worn Equipment and Emergency Egress Procedures. The Crew Worn Equipment lesson covers the Launch and Entry Suit and the rescue and survival gear. The Emergency Egress Procedures lesson covers the emergency egress procedures for pad egress, post landing and bailout situations. The combination of these two lessons provide the astronaut with on-orbit refresher training on all aspects of the Launch and Entry Suit, equipment and procedures required for emergency egress and escape. Prior to this project, this capability did not exist in one self-contained lesson. On the ground, these lessons will be used for familiarization training prior to advanced training in the simulators.

These lessons are interactive and integrate video, audio, graphics and text throughout the training flow. The Crew Worn Equipment lesson was also the first CBT lesson developed in both English and Russian. Another first was the embedded ability to configure the lesson to be taken pre-flight or in-flight. Those sections which are only applicable to pre-flight are not included in the lesson flow. A Review-a-Topic feature has been included so that the student does not have to take the complete lesson if they just want to review how to use the PRC-112 survival radio. These features can significantly reduce the in-flight time spent on refresher training. These two lessons were developed by the authors in 2.5 months and the finished product was delivered on a CD-ROM. The two lessons total over 325 megabytes of data, graphics, video, and audio.

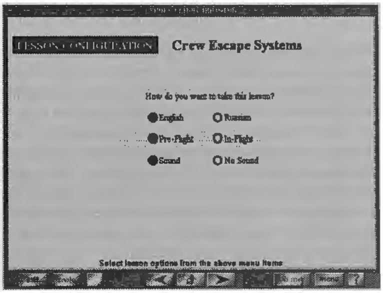
CBT Overview

The CBT lessons developed for use in astronaut training prior to these lessons had not used video or audio and their interaction was limited to question and answer sessions at the end of a lesson. The standard CBT was considered to be an electronic copy of the workbook and other printed material. This limitation may have been due to the fact that all previous CBT lessons had to be capable of running from a file server over a network. This limited the performance of any lesson developed. With the capability of the Crew On-orbit Support System (COSS) computer (2x CD-ROM drive, sound card and speaker) it would now be possible to run advanced multimedia CBT lessons in addition to commercial multimedia titles.

Macromedia Authorware 3.0 was the standard tool for developing CBT lessons in the Spaceflight Training Division when this project started. Macromedia is currently the industry leader for multimedia courseware development. At the start of this project, the authors of the this CBT had no previous experience with Authorware or multimedia productions. A great deal was learned in a very short time.

There were many unknowns when this effort started. The size of video window and frames per second the COSS platform would support. Which format video would be used Apple Quicktime video or Windows Audio/Video Interleaf (AVI). How audio would be used. These were all sorted out in the development process.

At the start of the project it was decided that the student would be able to configure certain options before beginning the lesson. The first option would be to configure the lesson for English or Russian due to the fact that Cosmonauts had been returned onboard the shuttle wearing U.S. launch and entry suits. Since this was a prototype project, a major goal was to demonstrate the CBT could be developed in multiple languages. This is an important concept since NASA has other international partners, (Japan, Russia, Europe, Canada). Another option the student is given at the start of the lesson is to have sound on or sound off. The user can also change the sound option at anytime during the lesson. The final choice the student has at the beginning of the lesson is to choose whether to take the lesson pre-flight or in-flight. Both lessons have English, pre-flight, and sound on as the default options (Figure 1). This lesson was designed to be taken by a first time user or as refresher training. If it is the users first time it is suggested that the complete lesson be taken and repeat any part as necessary. The lesson can be used as a refresher lesson because of the review a topic feature available to the user. In the review a topic section, a listing of all of the topics covered in the lesson is listed and can be accessed directly. At the completion of the topic the user is returned to the review a topic listing.



LESSON CONFIGURATION

Crew Escape Systems

How do you want to take this lesson?

● English ○ Russian

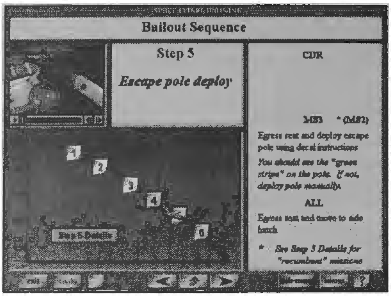
● Pre-Flight ○ In-Flight

● Sound ○ No Sound

Select lesson options from the above menu items

Figure 1

The procedures and cue cards are covered in the Emergency Egress Procedures lesson. One of the many new features developed for the lesson is 'hot spot clicking for more detail.' Many of the cue cards were too large for the complete procedure to be displayed on a computer screen. A hot-spot sensitive graphic is used to over come this limitation. The student clicks on the portion of the procedure of interest and an exploded view appears. This hot spot feature is used throughout the lesson. Another feature used is to 'click and drag' a shuttle icon to different points on a descent profile, representing multiple steps in a single procedure (Figure 2). This changes the text and video clip displayed on screen. Hypertext and hyperlinks were also used in the lessons. Hypertext words are blue and underlined, and the cursor changes to a hand when over the word. The user can click the mouse on the word and a box with a description or definition of the word will be displayed on the screen. The hyperlink is used on graphics and the cursor changes from an arrow to a hand. When the user clicks on the graphic, a box will appear on screen with more details or detailed photograph.



Bailout Sequence

Step 5

Escape pole deploy

CDR

MS3 *(MS2)

Egress seat and deploy escape pole using decal instructions

You should see the "green stripe" on the pole. If not, deploy pole manually.

ALL

Egress seat and move to side hatch

* *See Step 3 Details for "recumbent" missions*

Figure 2

Video clips are used throughout both lessons. The video clips have user controls located at the bottom of the clip providing standard VCR controls (pause/play, fast forward, rewind, slider bar) for Quicktime digital movies. The audio controls are not linked to the video. A decision was made to separate the video and audio because the video would be used in both the English and Russian versions. If the audio and video were linked together then two versions of each video would be required. This would double the space required on the CD-ROM as the video files are very large. Audio is used on most pages of the lesson to narrate text on the screen, similar to many commercial multimedia titles. This technique of audio narration of on-screen text gives multiple sensory inputs, both visual and aural, thus increasing training transfer rate and subject retention.

Student testing in previous CBT lessons used traditional question and answer techniques to evaluate knowledge retention. This format had received poor ratings from the CBT users. The 'knowledge challenge' was developed as a way to test the students' knowledge level by having them interact with graphics and photographs instead of the traditional text multiple choice questions. The knowledge challenge has the user click on the appropriate items or drag graphics in a certain order. It was felt that every effort should be made to hold the students' interest, and where possible entertain or amuse them.

Development Strategy

The opportunity to develop the CBT presented itself very late in the STS-76 timeline. From the time development started on October 16 1995, only 2.5 months were available until the finished product was required to be in bonded storage and ready for shipment to the Kennedy Space Center. This mandated a rapid applications development approach. The product was incrementally defined, developed and tested as the product matured.

Due to the short period of time allocated for development, the strategy was to utilize existing computer platforms, software tools, videos, pictures, graphics and the Crew Escape Systems training workbook. About 50% of the workbook text was typed in until an electronic copy was located. The electronic copy of the text was used to 'cut and paste' content into the lesson. Available resources were used to digitize the graphics and pictures. Video production facilities available at the Johnson Space Center digitized the video tapes. The authors provided the voice talent for the audio clips. A key advantage to using existing coursework material was the reduced time required for content verification by the subject matter experts.

Environment and Tools

The Simulator Operations and Technology Division's Stealth Lab was used to develop the CBT. The lab contained most of the necessary hardware and software tools required. Additional hard disks and a CD-Recorder (CD-R) were procured to support development. Macromedia Authorware was the multimedia development tool chosen to develop the lesson because it is the CBT development tool standard, and a copy was available in the Stealth Lab. The Spaceflight Training Division's standard authoring environment CBT template (modified for audio) was used to provide a consistent look and feel with other CBT lessons. Since neither of the authors had used Authorware before, an Authorware expert was assigned from the Spaceflight Training Division to provide support as required.

The following hardware and software tools were used during the development process.

Hardware

- 100 MHz Pentium PC with: 1 GB internal SCSI hard disk, 2 GB external SCSI hard drive, 32 MB RAM, Soundblaster, 4x CD-ROM, Pinnacle internal 2x CD-Recorder, PCI Diamond Stealth video card, 10BaseT Ethernet adapter, 20" color monitor
- 100 MHz Pentium PC with: 1 GB internal SCSI hard disk, 32 MB RAM, Soundblaster, 4x CD-ROM, PCI Diamond Stealth video card, 10BaseT Ethernet adapter, 20" color monitor
- Macintosh PowerPC 9500, 32 MB RAM, 1 GB hard drive, 4x CD-ROM, 20" color monitor
- HP ScanJet 3c
- HP LaserJet 4
- Epson Stylus Color inkjet printer
- Toshiba Satellite Pro 75 MHz Pentium multimedia laptop, 40 MB RAM, 850 MB hard disk, 4x CD-ROM, PCMCIA Ethernet adapter, 260 MB PCMCIA hard disk
- IBM ThinkPad 33 MHz 486, 12 MB RAM, 340 MB hard disk, multimedia docking station with 2x CD-ROM (Target platform identical to the flight unit)

Software

- Macromedia Authorware 3.0 - Authorware 3.0 provides an object-oriented, icon-based authoring environment with hypertext and multimedia capabilities, data measurement functions, and media integration controls. Its use of icons instead of scripting to define interactivity and control multimedia elements significantly shortens the learning curve for non-programmers.
- Corel PhotoPaint & Draw - Graphics image capture and editing software
- Canvas - Drawing software
- Paint Shop Pro - Screen capture software
- QuickEdit - Quicktime Video editing software
- SoundEdit - Audio capture and editing software

Development Process

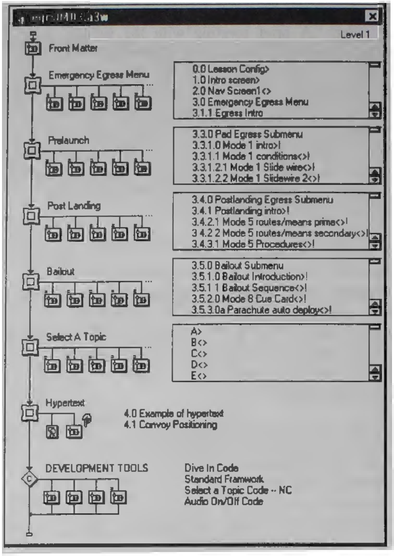
The first few weeks were spent gathering the existing training material, configuring the computers and ordering the additional hard disks and CD-ROM recorder. A brief meeting with the subject matter experts was held to get an idea of what the CBT should contain. The majority of the photographs were scanned in the first month, most of the video was edited in the middle of the project and the audio was recorded last. Learning Authorware was slow at first but a reasonable proficiency was obtained about 2-3 weeks into development. The local Authorware expert provided considerable help in learning the tool and explaining the advanced features..

A general concept was created for lesson organization and how the student would navigate through the CBT. This helped to decide what material to use and what video, graphic or audio media was required for the topic being covered. It also helped to identify missing media needed for lesson development. The next step was to take the materials and media and create the Authorware application.

Creating the Authorware Application

First, the Spaceflight Training Division standard CBT template was obtained. This template provides a standard look and feel (buttons, border, colors, navigation controls, menu templates, etc.) to the user. The existing template was not created with multimedia in mind and was modified to accommodate audio and video control.

Determining the paths of navigation through the lesson is an important aspect to consider when developing the application. Navigation in Authorware is something best learned by hands-on experience and could not possibly be covered here. Figure 3 shows the top level framework structure of the Emergency Egress Procedures lesson. The Emergency Egress Menu framework contains the lesson overview and pointers to major segments of the lesson. Each major segment (overview, pre-launch, post-landing, bailout and review-a-topic) is contained in a framework icon and broken into smaller pieces based on lesson flow. These smaller pieces are icons which represent a page or screen in the lesson. Referring to Figure 3, the page icons are the rectangular icons extending to the right of the square framework icon. Navigation options through the framework can be manipulated by the developer to control the students' path through the lesson.



0.0 Lesson Config>
1.0 Intro screen>
2.0 Nav Screen1<>
3.0 Emergency Egress Menu
3.1.1 Egress Intro

3.3.0 Pad Egress Submenu
3.3.1.0 Mode 1 intro>!
3.3.1.1 Mode 1 conditions<>!
3.3.1.2.1 Mode 1 Slide wire<>!
3.3.1.2.2 Mode 1 Slidewire 2<>!

3.4.0 Postlanding Egress Submenu
3.4.1 Postlanding intro>!
3.4.2.1 Mode 5 routes/means prime<>!
3 4.2 2 Mode 5 routes/means secondary<>!
3.4.3.1 Mode 5 Procedures<>!

3.5.0 Bailout Submenu
3.5.1.0 Bailout Introduction>!
3.5.1 1 Bailout Sequence<>!
3.5.2.0 Mode 8 Cue Card<>!
3.5.3.0a Parachute auto deploy<>!

A>
B<>
C<>
D<>
E<>

4.0 Example of hypertext
4.1 Convoy Positioning

Dive In Code
Standard Framwork
Select a Topic Code -- NC
Audio On/Off Code

Figure 3

Each page icon is actually a group of icons that make up the lesson content for that page. Developing each page is a simple yet involved process of dragging the desired icons from the toolbar or library onto the flowline and arranging them in the correct order. Icons dragged from the toolbar are blank, and text or graphical content must be added at this time. Navigation options and calculation icons must also be set up. Additional features, such as hotspots, buttons, animation, etc. should be used when appropriate to enhance the lesson and keep the student interested. Each page was tested in a standalone mode to verify correct operation. This is an oversimplification but provides some insight into the creation process.

Authorware provides the capability to create and use libraries to store audio, video and graphic icons. The use of libraries reduces the size of the main application file as only pointers or links to the library are used in the application. The effect is compounded if the same icon is used multiple times in the application. Separate libraries were created for each type of media. The audio and graphic libraries contains the actual audio or graphic file, whereas video libraries contain icons that point to external video files.

Media production

Video. The majority of the video clips used were taken from existing training video tapes. During development of the CBT, some video was captured during an ACES suit training session and used in the lessons. The videos were reviewed and segments selected for digitization. The video production facilities available at the Johnson Space Center were used to digitize the video tapes and store them in Quicktime format. The video clips were digitized then re-sized to the desired frame size and frame rate. The final step was to compress the video clips using Cinepak compression. This was a time consuming process as several of the video clips took almost 8 hours to compress. The compressed files were uploaded to a file server accessible to the development lab.

Final video editing was performed on a Macintosh 9500 Power PC using QuickEdit software. The files were downloaded from the file server, edited and saved in Quicktime Cinepak format. They were then transferred to the Authorware development platforms using the file server.

The video frame size, frame rate and compression format was a critical factor for the CBT. The target platform was an IBM Thinkpad 750, 33 MHz 486 laptop with 12 MB RAM and a 2x CD-ROM. Based on system performance, a target video bandwidth of 200 kbps was established. This translated to a frame size of 160 x 120 pixels at 15 frames per second. Even with this small picture size and frame rate, dropouts were experienced on the target. New CD-ROM drivers were installed which fixed the problem and even provided sufficient throughput to allow a larger frame size of 240 x 180 to be used at the same frame rate.

Audio. Audio clips were recorded and edited on the Toshiba laptop using SoundEdit software. The built-in microphone on the laptop was initially used but the audio quality was poor. A studio quality lapel microphone was acquired and provided greatly improved sound for the audio clips.

Graphics. Numerous graphics files were produced during the development process. Several intermediate files were created for every final graphic file used. All of the final images and most of the scanned master images are available on the CD-ROM for viewing outside the lesson. Corel

PhotoPaint was the software tool used for the graphic scanning and editing process. PaintShop Pro was also used to capture on-screen graphic images from the workbook.

The process used on existing photographs was to scan the picture and store it as a high resolution (24-bit color) JPEG file. This master image was then cropped, re-sized and converted to 256 colors (8-bit) using the Error Diffusion Dithering and Optimized Palette options in PhotoPaint. To improve picture quality, an Adaptive Unsharpen process was performed, if required, and the file saved. This image was usually the final version unless additional processing was required, such as changing the background color.

CD Production. CD production is actually a very simple process, once you read the manual. The CD recorder (CD-R) came with software and it was fairly easy to use. First, all the files to be saved are copied on to the PC with the CD-R. Then a 'virtual image' of the CD to be recorded is created using a tool that looks similar to Windows File Manger. From this point, a ISO-9660 disk image can be created on the hard drive or the CD can be created directly from the virtual image file list. Without getting into the details, it is best to create the ISO-9660 image and 'burn' the CD from this image.

The final step in producing the CD-ROM was to create the jewel case inserts and CD label graphic. This can be done using any graphics program. Corel Photo Paint and Corel Draw were used once again. The Epson color printer produced excellent inserts at 720 dpi for the development releases. Inserts for the production run CDs were printed by the CD-ROM manufacturer using a 4 color process for the inserts and 2 color for the CD label.

Testing

There were three stages of testing. The first stage was to run the lesson while developing in Authorware (from the start or just certain pages) to test navigation, page execution and overall performance. The second stage was to compile the lesson and execute it on the development platform. The last stage was to burn a CD-ROM and test the application on the target platform. The development platform or the Toshiba laptop were used if the target was unavailable. The target was always the final platform for evaluating performance.

CBT Launchpad

The finishing touch was a small application written in Delphi that is used to select and execute the CBT lessons. The CD is Windows 95 autorun capable and will automatically start the launch program when the CD is inserted into the CD-ROM drive. It can also be run from the Windows File Manager.

CBT Deployment

Since this CBT was targeted at supporting LDM-2, the expected production run was around 10-15 CD's. After demonstrating the lessons to the subject matter experts, training instructors and several astronauts, it was apparent that a larger quantity would be required. The instructors thought the CD-ROM would be a great compliment to, if not replacement of, the training workbook and videos currently used. It also appears the CD will be a standard item on the remaining LDM missions aboard Mir. Several CD-ROM manufacturers were contacted and cost estimates were obtained. A large order was contracted and these will be distributed among the NASA astronaut and training community.

Lessons Learned

The most classic of lessons learned applied to this development... Save your work often and create daily back-ups on a separate machine or file server! Fortunately, both were being done on this project. Either due to software incompatibilities or the computer hardware, there were several occasions where the machine would lock-up and reboot. Most of the time the files were undamaged, but on a few occasions, the source files were corrupted and had to be restored from the most current back-up. It didn't take as much time to redo the work as it did to initially create it, but it was still wasted time.

Editing video requires a lot of hard disk space. This was quickly observed when a relatively small uncompressed video file consumed 200 MB of disk space. The only solution available was to use the fileserver until the files could be edited and saved in compressed format. New hard drives were ordered which helped immensely.

The AVI video format was initially selected for the video files. Unfortunately, at the time this CBT was developed, drivers were not available for Authorware that would play AVI files and provide the user with VCR-type controls over the video. A driver is available for Quicktime video files that provides this capability.

An important thing to consider when digitizing the video is the frame rate, frame size and format. Choose these parameters based on the target platform and the tools being used to create the product. A large picture playing at a high frame rate will require significant bandwidth and CPU resources. The three sizes of video were available (160x120, 240x180, and 320x240) with a frame rate from 1 to 30 fps. A 160x120 frame with a rate of 15 fps provided the best compromise between quality of video motion and transfer rate from the CD-ROM player. The frame size was later increased to 240x180 when improved drivers were installed on the target platform.

A valuable lesson learned when cutting and pasting MS Word files into Authorware is not to use the 'Edit | Paste Special | Word 6.0 Document' method. Although this seems like an attractive method to import text and graphic and retain the original formatting, it's not that great when you find out how many bytes it takes Authorware to store it. Whether text, graphics, tables or whatever, Authorware stores the imported item as a graphic image. At one time, the procedures lesson was up to 60 MB of source file!

This is a recommended way to import text from Microsoft Word. While editing the document in Word, group the text that you want to appear on an Authorware page by placing Page Breaks in the appropriate locations. Save this file as an RTF formatted file. Exit Word and go to Authorware. Drag a Map icon somewhere in the flow and open it. Drag a Display icon onto the flow of the Map and open it. Import the word file using the 'File | Import' method. It will store each Word page in a separate Display icon within the Map icon. From there, you will have to edit the text to get it the way you want it. It is better than typing it over again.

An increase in productivity was perceived on this project. This is believed to be the result of being assigned full-time (for the most part) to the project versus part-time (working multiple tasks). The authors worked very long hours (usually 10-12 hour days, and sometimes 16 hours or longer during final development and integration). Breaks were taken for lunch, dinner, etc. but the continuity and lack of interruptions probably contributed to an increase in productivity.

A major lesson learned from this project was that the developer does not have to be an expert on the subject nor experienced with the development tools. Neither author is an expert or instructor on the Crew Escape Systems, and neither had experience with Authorware before this project. It is important to be able to learn the tools quickly, devote time to learning the tools and not be afraid to ask for help.

Development Metrics

Activity	Total
Authorware Development	389.5 hours 6854 icons
Authorware Consulting	40.0 hours
Audio Production	41.5 hours 194 files 64.1 MB
Graphics Editing	55.0 hours 408 files 112.3 MB
Video Capture/Editing	66.0 hours 64 files 147.8 MB
Video Production	196.65 hours
SME Review & Validation	12.0 hours
CD-ROM Production	35.0 hours
Computer Support	60.0 hours
Total	6854 icons 895.6 hours 666 files 324.2 MB

It must be noted that the hours above should be interpreted for activities that were not performed during the development process. Specifically, because existing approved material was used, it was not required that the content be developed from scratch. The original content was available and used, although it was reworked considerably for the CBT lessons. The video hours charged on the project are a little high due to the 'learning' that took place in the process. The video production hours could be reduced in future efforts. One last thing to note is the total icon count in the table includes the standard CBT template icons. The actual number of icons developed by the authors is 6414.

Future Activities

In-flight training will become even more significant when the Space Station begins operating. With missions lasting 3 months or longer, an astronaut may not begin an experiment until the third month. Refresher training on the payload may be of great benefit in these cases.

Several new techniques and technologies will be explored over the next few years under the Field Deployable Trainer project. Ideas already in work include training across the internet using Web-based browsers. An independent uplink/downlink capability using commercially available equipment is being explored. For several situations, such as 'just-in-time training', CBT applications can be created and uplinked to a training platform on-orbit. The future is bright for CBT technology development.

Conclusions

These lessons were developed with the goal of providing training "anytime-anywhere" and to show the potential for future in-flight training. Constraints on the human space flight program mandate that this will have to be the philosophy of training in the future. In the past, the training for space flight has been in large costly facilities. The astronauts' time has always been a limited resource. As Space Station training is combined into the training flow, the astronauts' time for training will be even more scarce. Using the approach of interactive multimedia computer based training, the student can take refresher training, just in time training or ground-based training lessons as time permits or as the situation requires. For example, a CD-ROM could be sent to a Payload Specialist in Europe and they can start their training before arriving at NASA-JSC for the final phase of integrated training. These lessons have been mass produced on CD-ROM for about $1.50 each and can be distributed to any user who has a multimedia PC running Windows - a very cost effective way to provide training.

AIAA-96-3500-CP

AN OBJECT-ORIENTED IMPLEMENTATION OF RIGID BODY SYSTEMS IN SPACE TRAINING SIMULATION

John E. Sellke, Principal Software Engineer
Hughes Training, Inc., Houston, Texas 77058
and
Danny Shumock, Senior Associate
Booz-Allen & Hamilton, Inc., Houston, Texas 77058

This paper presents a unique, object-oriented software design for the dynamics simulation of spacecraft which are treated as systems of rigid bodies. The coupling introduced in traditional approaches to rigid body system dynamics greatly reduces the flexibility of a software implementation. Therefore, equations are derived using the concept of relative angular momentum that completely decouple the effects of attached components from their parents. The software architecture is an adaptation of object-oriented techniques developed by Booch, Coad-Yourdon, Rumbaugh, and Shlaer-Mellor to large-scale real-time applications. The results, from an Ada implementation of this method in the Space Station Training Facility, are presented and compared to a traditional dynamics model.

Introduction

The International Space Station (ISS) is a highly complex space vehicle which is being co-developed by governmental agencies from around the world under the leadership of the United States National Aeronautics and Space Administration (NASA). The ISS consists of numerous articulated components, such as robotic manipulators, a mobile transporter, and rotating truss segments with attached thermal radiators and photovoltaic arrays. These components are assembled on-orbit over time, using different kinds of launch vehicles.

The Space Station Training Facility (SSTF) located at Johnson Space Center in Houston, Texas will train astronauts in the assembly, operation, and maintenance of the ISS. In particular, SSTF training simulations will develop the hand-eye coordination needed for proximity and robotics operations. The dynamic characteristics of the ISS components, coupled with training requirements for: (1) rapid reconfiguration of executable loads without software code changes and (2) real-time execution, present a unique challenge in the simulation of vehicle dynamics.

State-of-the-art robotics software and computer hardware technology allow for the real-time simulation of multi-segmented manipulators treated as a set of rigid or flexible bodies. A dynamically reconfigurable simulation of several such manipulators attached to an unconstrained basebody which *itself* has articulated components is not feasible using current techniques, given the requirements constraints.

A new simulation approach which satisfies these requirements has been developed in the SSTF; it is based on concepts borrowed from visual system image generator technology. In a visual scene, a space vehicle is represented by a single-ended tree structure of chained coordinate systems. This abstraction allows a tree node (vehicle component) to be connected to, or disconnected from, another node. Also, a node may be moved or rotated independently and in concert with its parent node, whenever the parent is moved or rotated.

The abstraction is easily extended to compute the composite mass properties at each node. However, an extension to support vehicle dynamics is more difficult to achieve—that effort requires the derivation of equations of motion which incorporate the effects of "internal" component motion.[1] The derivation of these equations entails a review of Euler's original work in rotational dynamics, which includes an examination of the definition and relationships of torque, angular momentum, and inertia. In so doing, it is possible to separate the equations of motion of each component into two parts: (1) the contribution from the motion of the component relative to its parent's frame of reference and (2) the contribution from the transformed and transpositioned inertia and angular momentum of the *subtree* of components which are attached to the component. The resulting system of equations is similar in form to those which model control moment gyroscopes and momentum wheels.

In the SSTF, the dynamic motion of each robotic manipulator is computed using classical techniques; each manipulator is then coupled to the ISS basebody using the new approach. The equations of motion are implemented in the Ada programming language using object-oriented software design techniques.

Rotational Equations of Motion

This section of the paper presents a derivation of the rotational equations of motion (REOM) for a space vehicle which incorporates the effects of its internal articulated component movement.

The rotational equations of motion of a space vehicle considered simply as a rigid body are given by:

$$\sum_i \vec{\mathbf{N}}_i^{ext} = \dot{\vec{\mathbf{L}}}_V , \tag{1}$$

where $\sum_i \vec{\mathbf{N}}_i^{ext}$ is the sum of the external moments about the vehicle center-of-mass. Equation (1) can be expressed in an inertial frame of reference (such as True-of-Date) by:

$$\sum_i \vec{\mathbf{N}}_i^{ext} = \left(\dot{\vec{\mathbf{L}}}_V\right)_r + \left(\vec{\omega} \times \vec{\mathbf{L}}_V\right), \tag{2}$$

where $\left(\dot{\vec{\mathbf{L}}}_V\right)_r$ is the time derivative of the vehicle angular momentum as seen by an observer rotating with the vehicle (hence the subscript r). Noting that $\vec{\mathbf{L}}_V = I\vec{\omega}$, where I is the vehicle inertia tensor at the center-of-mass and $\vec{\omega}$ is the vehicle body rate vector, (2) becomes:

$$\sum_i \vec{\mathbf{N}}_i^{ext} = I\dot{\vec{\omega}} + \dot{I}\vec{\omega} + \left(\vec{\omega} \times I\vec{\omega}\right). \tag{3}$$

In general, however, a space vehicle has articulated components and therefore must be treated as a *system* of connected rigid bodies. The motion of these components creates "internal" angular momentum which must be dealt with in the REOM of the vehicle. We account for this by adding an additional term to (1):

$$\sum_i \vec{\mathbf{N}}_i^{ext} = \dot{\vec{\mathbf{L}}}_V + \dot{\vec{\mathbf{L}}}_V^{int} . \tag{4}$$

In an inertial frame of reference, (4) becomes:

$$\begin{aligned}\sum_i \vec{\mathbf{N}}_i^{ext} = \left(\dot{\vec{\mathbf{L}}}_V\right)_r &+ \left(\vec{\omega} \times \vec{\mathbf{L}}_V\right) \\ +\left(\dot{\vec{\mathbf{L}}}_V^{int}\right)_r &+ \left(\vec{\omega} \times \vec{\mathbf{L}}_V^{int}\right).\end{aligned} \tag{5}$$

Substituting $I\vec{\omega}$ into (5) for $\vec{\mathbf{L}}_V$ and transposing $\left(\dot{\vec{\mathbf{L}}}_V^{int}\right)_r$, we obtain:

$$\begin{aligned}\sum_i \vec{\mathbf{N}}_i^{ext} - \left(\dot{\vec{\mathbf{L}}}_V^{int}\right)_r &= I\dot{\vec{\omega}} + \dot{I}\vec{\omega} \\ &+ \left(\vec{\omega} \times I\vec{\omega}\right) + \left(\vec{\omega} \times \vec{\mathbf{L}}_V^{int}\right).\end{aligned} \tag{6}$$

We treat the $-\left(\dot{\vec{\mathbf{L}}}_V^{int}\right)_r$ term like another external moment which must be passed to the REOM and we denote this moment by $\vec{\mathbf{N}}_V^{int}$. The presence of the additional term $\left(\vec{\omega} \times \vec{\mathbf{L}}_V^{int}\right)$ in the REOM represents the *stiffness* of the vehicle to changes in its angular velocity due to the angular momentum of its articulated components. Solving (6) for $\dot{\vec{\omega}}$ and integrating yields:

$$\begin{aligned}\vec{\omega}(t+\Delta t) = \int_t^{t+\Delta t} I^{-1} \Bigg[&\sum_i \vec{\mathbf{N}}_i^{ext} + \vec{\mathbf{N}}_V^{int} \\ &- \dot{I}\vec{\omega} - \left(\vec{\omega} \times I\vec{\omega}\right) - \left(\vec{\omega} \times \vec{\mathbf{L}}_V^{int}\right) \Bigg] dt ,\end{aligned}$$

where $\vec{\omega}(t+\Delta t)$ is the vehicle body rate vector at time $t+\Delta t$.

We approximate $\vec{\mathbf{N}}_V^{int}$ and $\dot{I}$ with:

$$\vec{\mathbf{N}}_V^{int} \approx -\frac{\vec{\mathbf{L}}_V^{int}(t) - \vec{\mathbf{L}}_V^{int}(t-\Delta t)}{\Delta t}$$

and

$$\dot{I} \approx \frac{I(t) - I(t-\Delta t)}{\Delta t} .$$

Note that these approximations introduce a lag into the REOM. We can compensate for this by extrapolating $\vec{\mathbf{N}}_V^{int}$ and $\dot{I}$ to $t+\Delta t$; however, since space vehicle components move so slowly, the lag is correspondingly small and can be ignored.

We are now prepared to discuss a method for solving the REOM. We seek an algorithm for computing the total angular momentum of the vehicle articulated components, $\vec{L}_V^{int}(t)$, since it is clear that $\vec{\omega}(t+\Delta t)$, i.e., the REOM solution, is dependent upon it. The algorithm will be presented in two stages: we introduce an abstraction for the vehicle first and use it to discuss overall processing concepts, and then conclude with the physics of the algorithm.

We simulate the rotational dynamics of a space vehicle by treating it as a system of connected rigid bodies (components) which can move relative to each other. Each component of the vehicle is represented by a *node* in a hierarchical data structure—a *tree* (denoted by T_V and shown in Figure 1). The *root* of the tree represents the vehicle as a whole and is also a node (shown as a dashed circle in the Figure), but the vehicle itself is not considered to be a component—it is a collection or aggregate of components. The topology of T_V reflects the real-world connectivity of the vehicle components, and can be changed during real-time execution; at no time, however, does the tree ever become a cyclic graph (a 1-dimensional structure with closed loops). The connectivity of the tree is described in terms of *parent-child* relationships between the nodes.

The *subtree* of a component C_i is the set of those components which are the children of C_i, the children of the children of C_i, the children of the children of the children of C_i...until there are no more children. Note that C_i is not included in its subtree. For example, in Figure 1, the subtree of component C_2 (denoted by S_2) is the set of components

$$S_2 = \{C_7, C_8, C_9, C_{10}, C_{15}, C_{16}, C_{17}\}.$$

Similarly, $S_8 = \{C_{15}, C_{16}, C_{17}\}$ and $S_1 = \varnothing$. A recursive definition of a subtree is given by the following: the subtree of a component C_i is the set of those components which are the children of C_i *and their subtrees*. The subtree of a component moves with that component *as a whole*, so that if a component translates or rotates, its subtree translates or rotates with it; this motion constitutes the "external" motion of the subtree. The "internal" motion of a subtree arises from the motion of its individual components plus the external and internal motion of their subtrees.

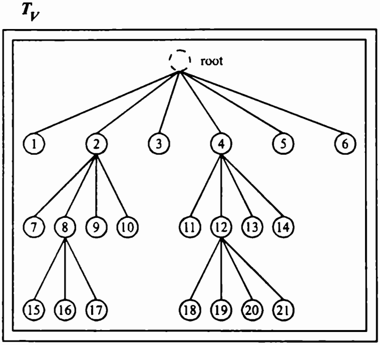


Figure 1. Typical Space Vehicle Tree

Each component in the tree has a *configuration state*. The configuration state of the component C_i consists of the following attributes:

- Parent
- Number_Of_Children
- Configured_Status
- Attached_Status

The `Parent` attribute indicates which other component in the tree is the parent of C_i; the parent of the root is the root. The `Number_Of_Children` attribute, if >0, indicates that C_i is itself a parent with N number of child components. The `Configured_Status` attribute indicates whether or not C_i is present in the simulation and should be included in the REOM computations; this attribute facilitates the reconfiguration of the vehicle as its components are added or deleted over time. The `Attached_Status` attribute indicates whether or not C_i is currently attached to its parent and should be included in the REOM computations.

Each component in the tree has a *motion state* whose attributes can be arranged in two distinct groups: dynamics attributes and kinematics attributes. The dynamics attributes of the component C_i consist of the following values:

- Component Mass (m_{C_i})
- Component Center-of-Mass ($\vec{\mathbf{c}}_{C_i}$)
- Component Inertia Tensor at $\vec{\mathbf{c}}_{C_i}$ ($\left[I_{C_i}\right]$)
- Component Internal Linear Momentum ($\vec{\mathbf{p}}_{C_i}^{int}$)

- Component Linear Momentum ($\bar{\mathbf{p}}_{C_i}$)
- Component Internal Angular Momentum ($\bar{\mathbf{L}}_{C_i}^{int}$)
- Component Angular Momentum ($\bar{\mathbf{L}}_{C_i}$)
- Subtree Mass (m_{S_i})
- Subtree Center-of-Mass ($\bar{\mathbf{c}}_{S_i}$)
- Subtree Inertia Tensor at $\bar{\mathbf{c}}_{S_i}$ ($\left[I_{S_i}\right]$)
- Subtree Internal Linear Momentum ($\bar{\mathbf{p}}_{S_i}^{int}$)
- Subtree Linear Momentum ($\bar{\mathbf{p}}_{S_i}$)
- Subtree Internal Angular Momentum ($\bar{\mathbf{L}}_{S_i}^{int}$)
- Subtree Angular Momentum ($\bar{\mathbf{L}}_{S_i}$)

All of the dynamics attributes (except the masses) are defined relative to a coordinate system which is attached to the component, i.e., the *local* frame of the component. The position of the local origin is arbitrary, but it is usually fixed on the basis of geometric symmetries that the component may possess. The local frame is *not* the component's body frame of reference. The origin of the body frame is located at the component's center-of-mass, but the local frame is fixed at some *geometric* reference point of the component. The values of $\bar{\mathbf{p}}_{C_i}^{int}$ and $\bar{\mathbf{L}}_{C_i}^{int}$ are zero unless C_i is represented by an instance of another tree (see below).

The kinematics attributes of the component C_i consist of the following values:

- Component Position ($\bar{\rho}_{C_i}$)
- Component Velocity ($\bar{\mathbf{v}}_{C_i}$)
- Component Attitude Quaternion ($\hat{\mathbf{q}}_{C_i}$)
- Component Attitude Matrix ($\left[\hat{\mathbf{q}}_{C_i}\right]$)
- Component Angular Velocity ($\bar{\omega}_{C_i}$)

$\bar{\rho}_{C_i}$ and $\bar{\mathbf{v}}_{C_i}$ represent the position and velocity of the component frame relative to its parent, expressed in the parent frame. $\mathbf{q}_{C_i}$ transforms a vector from the component frame to the parent frame. $\bar{\omega}_{C_i}$ is the body rate vector expressed in the component frame.

From the space vehicle problem space we abstract a MassPersp class.* The MassPersp class encapsulates the internal behavior of any *set* of connected rigid bodies using the data structures and states we have just presented. An instance of the class is made by specifying an enumerated list of component names and the name of the aggregate object. The class provides: modifiers for the configuration state of each component; modifiers for a subset of the motion state attributes (mass, center-of-mass, inertia tensor, internal linear momentum, and internal angular momentum) of each component; selectors for all of the component and aggregate states; and methods for the computation of the dynamics attributes of each subtree and the aggregate.

The calculation of $\bar{\mathbf{L}}_V^{int}(t)$ begins with updates to the component states using the MassPersp modifiers. The update values are generated by objects which encapsulate the dynamic behavior of each individual component. When the updates are complete, the component tree is traversed, beginning at the deepest level of the tree and progressing recursively upward until the root is reached. All of the mass properties attributes are computed in the first. Then, using those results, the remaining dynamics attributes are calculated where each individual component and every subtree contributes angular momentum to $\bar{\mathbf{L}}_V^{int}(t)$. $\bar{\mathbf{L}}_V^{int}(t)$ is then passed to the REOM for final solution.

The MassPersp class is an abstract data type that can be reused, for example, to treat a component which is already part of a MassPersp object as *another* MassPersp object (see Figure 2). In the Figure we have a MassPersp object labeled T_V which represents a space vehicle; C_i is one of the components of T_V. We also have another MassPersp object labeled T_{C_i} which represents the component C_i. Now, there are two ways to interpret T_{C_i}: either, (1) it represents part of the subtree of the parent of C_i, or (2) it represents the "inner workings" of C_i which is considered to be a "black box". In either case, we allow *components* to possess "internal" linear and angular momentum which are calculated by their representative objects. In the Figure, both of the internal momentum attributes of C_i are calculated in T_{C_i} first, then are sent via a an *object message* to T_V in the form of updates to C_i. This particular message between the objects can be viewed as the restriction of an attribute mapping $f: T_{C_i} \rightarrow T_V$, where

$$f\left(m_{root}\right) = m_{C_i} \ ,$$

[*] See next section. The full rotational behavior of a space vehicle is actually encapsulated in a derived instance of this class.

```
MT (ft)  : 0.0, 100.0, 0.0
MBS(ft)  : 0.0, 0.0  , 1.87
Cmd  :  SHY joint
Arm config.(deg): 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
```

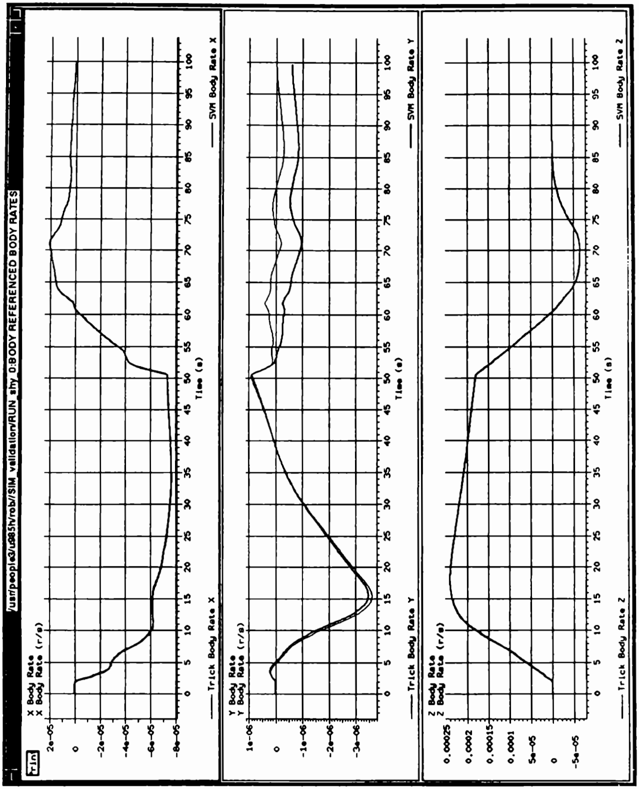


Figure 7. ISS Body Rate Vector Results (Test #2)

rate vector using the new algorithm against a high-fidelity dynamics model called Trick. Trick was developed by NASA, primarily for doing engineering studies on the Space Station Remote Manipulator System (SSRMS). The new algorithm was executed at 20 hertz and Trick was executed at 2000 hertz.

All of the tests use a nominal ISS configuration which includes the Mobile Transporter (MT), the Mobile Remote Servicer Base Structure (MBS), and the SSRMS. The MT is a platform that translates along the ISS truss and is used to position the SSRMS for assembly and servicing operations. The MBS rides on the MT and the shoulder segment of the SSRMS is attached to the MBS.

In the first test, the MT is initially positioned 100 feet along the starboard ISS truss and the SSRMS shoulder is pitched up 90° (all other joint angles are initialized to zero). The shoulder is then pitched down and the effects of that motion on the ISS body rate vector is calculated and plotted. The third test repeats the same shoulder motion as in the first test, except that the MT is initially placed at the ISS structural origin.

In the second test, the MT is initially positioned 100 feet along the starboard ISS truss and all of the SSRMS joint angles are set to zero. The shoulder is then yawed in a positive sense and the effects of that motion on the ISS body rate vector is calculated and plotted. The fourth test repeats the same shoulder motion as in the second test, except that the MT is initially placed at the ISS structural origin.

The plot in Figure 7 displays the results from the second test, which was considered to be the worst-case test of the group. The plot clearly shows that the new algorithm calculated the ISS body rate vector in excellent agreement with Trick.[‡]

Conclusion

Software has been developed in the SSTF which implements a generalized object-oriented solution to the space vehicle dynamics problem. The implementation can be reused to simulate the dynamics of any kind of space vehicle from telecommunications satellites to interplanetary spacecraft. The software can be rewritten in Ada 95, C++, or any other object-oriented programming language and targeted on other computational platforms. The results obtained from this implementation compare favorably with other industry approaches, and uses significantly less computing resources. Analysis of the reconfigurability and reusability of the implementation indicate a lower life-cycle cost for the SSTF, and the potential for cost savings on other projects.

References

[1]Greenwood, D. T., *Principles of Dynamics*, Second Edition, Englewood Cliffs, NJ: Prentice-Hall, 1987.

[2]Booch, G., *Object-Oriented Design*, Redwood City, CA: Benjamin-Cummings, 1991.

[3]Coad, P., and Yourdon, E., *Object-Oriented Analysis*, Second Edition, Englewood Cliffs, NJ: Yourdon Press, 1991.

[4]Rumbaugh, J., *Object-Oriented Modeling and Design*, Englewood Cliffs, NJ: Prentice-Hall, 1991.

[5]Shlaer, S., and Mellor, S. J., *Object-Oriented Systems Analysis*, Englewood Cliffs, NJ: Yourdon Press, 1988.

[‡] In the plot, the data produced by the new algorithm is identified as "SVM Body Rate {X|Y|Z}".

FIELD DEPLOYABLE TRAINER PROJECT

Sean M. Kelly[1], David B. Chiquelin[2] and Ronald B. Lee[1]

[1]Mission Operations Directorate
NASA Johnson Space Center
Houston, Texas

[2] Spaceflight Training and Facility Operations
Support Systems Operations
United Space Alliance

Abstract

The Field Deployable Trainer (FDT) project was conceived as a way for Astronauts to train anytime and anywhere. Historically, Astronauts receive most of their training at the Johnson Space Center (JSC) with additional training conducted at other locations around the world, requiring significant amounts of crew travel. Training is even more distributed with the International Space Station (ISS). ISS mission lengths can be 90 days or longer with in-flight skill and knowledge degradation a possibility. Logistics flights may also bring experiments on which crewmembers have not been trained. The nature of the ISS warrants some type of in-flight training. The goal of the FDT is to fill these needs by providing, in a portable platform, simulations and technology previously unavailable to Astronauts in the field or inflight. The three year prototyping phase will test and develop new methodologies, determine degradation curves, and provide a management structure for the operational phase. The first generation of FDT prototypes went from a concept stage in late July 1995 to a completed, space-flight certified system by January 1996. Currently onboard Mir, the multimedia capabilities are used for training as well as psychological support. The interface collects information on applications usage, to be analyzed post flight.

Project Background

The idea of onboard training was considered by this team in 1988 as part of the original operations assessments for the Extended Duration Orbiter (EDO) program. The goal of the EDO program was to extend the on orbit capabilities for the shuttle orbiter to as long as 30 days. As part of the operational assessment, the concern was raised on crewmember ability to land the orbiter after 25-30 days. The concern focused on two areas: crewmember proficiency and physical degradation. Physical inflight degradation is the subject of ongoing space life sciences investigations.

Since no data existed on the area of crewmember proficiency, an informal survey of flown astronauts was conducted, with surprising results. Some crewmembers felt some type of skill degradation occurred in psychomotor and cognitive skills even after the relatively short seven day shuttle flights. The concerns were significant enough that serious consideration was given to upgrading the orbiter to full autoland capability. The alternative was to provide embedded in-flight training, a costly option at best. Eventually the EDO program was reduced to support mission lengths of approximately 14 days, but the concerns raised on crewmember skill proficiency levels remained operationally unaddressed.

The Space Station Freedom (SSF), program started transitioning from a concept level towards implementation during 1991. At this time, the Mission Operations Directorate became more involved in the design of SSF, and included in the baseline operational plan the use of On Board Training (OBT). As the SSF was redesigned into the International Space Station (ISS), the role of OBT was strengthened, not weakened.

For ISS a complete review of how, where, when and why, as well as the costs involved, was accomplished. It became clear as part of this zero-base review that the traditional methods have significant cost issues, scheduling issues, and availability issues. Clearly, the traditional methods would not only be cost ineffective and result in unacceptably long preflight training templates, in many scenarios they would be physically impossible to accomplish.

The traditional methods had been designed to support short flight durations, with very tight in-flight schedules. Due to the duration of shuttle flights the cost per hour of flight time is very high, making it both cost effective and mission success practical to overtrain a flight crew preflight. However, with the ISS, a new paradigm for training is required, which is reflective of the mission duration, relaxed in-flight scheduling, and budget availability.

The focus of the traditional methods of training delivery for core systems has been to use part task and full task training facilities. This methodology has served the human space program well, but has an associated high cost. As an example, the Shuttle Mission Simulator has over 300 people associated with facility and software sustaining engineering. Training costs for this facility are between eight thousand and twelve thousand dollars per hour.

For shuttle payload and experiment training, crewmembers currently travel to Principal Investigator (PI) sites, as well as manufacturing facilities. This requires a significant amount of travel, and quite often occurs close to flight, requiring the crew weekly workloads to significantly increase 2 to 3 months prior to launch. Fatigue and training retention are constant concerns.

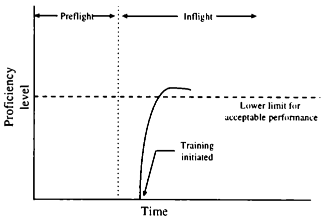
For the ISS, the distribution of training facilities and complexity of generating training flows will drastically increase. Using the traditional paradigms, payload, experiment and core systems training will occur in Russia, Canada, Japan, France, Italy, and Germany. As a complicating factor, in many instances the International Partner (IP) organizations have very limited experience with the current training techniques and standards used by NASA. The responsibility for indoctrinating these IP organizations will fall to the Spaceflight Training Division at the Johnson Space Center. Philosophies, methods and standards that have been used, and new ones that are being developed for the ISS, will have to be documented to a level previously avoided in order to overcome cultural and language barriers that exist. Also, in some instances, the simulator models for core systems will not be available in time for astronauts to receive prelaunch training on systems that they will install inflight.

Two types of Field Deployable Training are being developed to support human space flight: preflight ground-based remote and On Board Training (OBT). Ground-based remote training would focus primarily on maintaining skill and knowledge levels for crewmembers while they are away from the primary training site. In the currently drafted training plans crewmembers will be at remote locations for many months. By utilizing ground-based remote training to maintain proficiency levels, costly retraining can be avoided.

For ISS, three areas of On Board Training have been defined: Just In Time Training (JITT), Proficiency Maintenance and Refresher. All three areas have been accepted at the Concept Of Utilization level, and require further definition and testing prior to implementation for ISS.

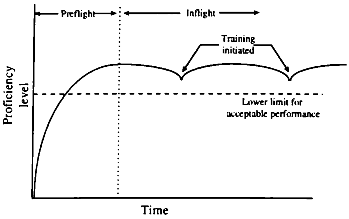
Just In Time Training is currently defined as new training for a task that occurs just prior (within 24 hours) to the planned execution time of the task. This area would include unanticipated tasks, such as in flight maintenance, and low criticality/low risk tasks which the astronaut had received only familiarization level training on preflight. JITT for unanticipated tasks will heighten the probability of mission success. In the case of low criticality/low risk tasks, the advantage of JITT is the ability to significantly off-load the congested prelaunch training template and provide better training retention. Since the training received is very close to the time of task execution, task/knowledge conflict is reduced and recency is presumed to be much higher.

Just In Time Training



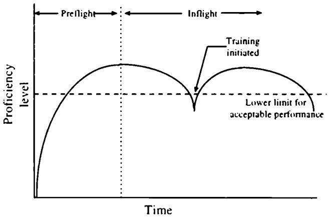
Proficiency Maintenance Training (PMT), as currently defined, would be conducted for high criticality tasks or skills on a standard recurring basis. Evaluation would be embedded into the training material for the purpose of determining the student proficiency levels, adjustment of PMT timeframes, and identifying any possible deficiencies with the training material itself. PMT requirements can be filled by actual task execution, in lieu of structured training.

Proficiency Maintenance



Refresher training is defined as training that would occur if the crewmember's performance had degraded to a level below established acceptable standards. This type of training would be conducted at either the discretion of the crewmember or as a result of deficiencies identified by PMT. At the discretion of the crew commander, flight director or training supervisor, the refresher training requirement can be waived on a case by case basis and will be dependent on task criticality.

Refresher Training



In order to convert the ISS concepts into practical operational products many issues had to be addressed. Significant funding would be required during the operational phase of ISS, and a detailed needs assessment would be required. Since psychomotor, cognitive and affective domain degradation curves had never been documented with regard to space flight, they would need to be developed and evaluated. Additionally, the logistics required to support cost effective, rapid training material development and sustaining engineering do not exist and need to be determined and developed.

In an effort to accomplish the above the Field Deployable Training project was initiated. Funded by NASA Headquarters research funds, the FDT project began in earnest in May 1995.

Project Goals

The goals of the FDT project in support of ISS are to refine the concepts for OBT and ground based remote training, to generate initial cognitive, psychomotor and affective domain degradation curves, determine the initial proficiency maintenance OBT requirements, reduce training costs, and establish the logistics to support FDT delivery.

These goals will be achievable in a cost effective manner, provided that non-traditional approaches are taken. One example is using a very small development team with wide ranging experiences. The development team has also taken the controversial stance that one method of significantly reducing the training load for an astronaut is to provide better real time support products. Another example is the aggressive use of commercial off the shelf (COTS) products wherever possible, for both development tools and for user applications. While NASA unique software has been the norm for human space flight, this was originally done because of the lack of quality in COTS products, as well as NASA unique requirements. In both cases, the gap is closing.

With the widening astronaut experience base, astronauts are now more accepting of COTS software tools, and in many instances have previous experience with them. In the area of supporting general reference material, the quality of the commercial products exceeds that which could reasonably be produced in-house, at a much lower cost.

Initial On-orbit Field Deployable Trainer The Crew On-orbit Support System

In response to stated astronaut concerns from the first flight of a NASA astronaut onboard the Russian space station Mir, the FDT team was requested to develop a Crew On Orbit Support System (COSS) in time for the next NASA mission to Mir. The Phase 1 Program Office, which administers the NASA-Mir program, felt that NASA could do a better job at supporting astronauts inflight. A Tiger Team of experts in training, psychological support, logistics and flight hardware development, as well as Phase 1 Office representatives, was formed to establish baseline requirements and obtain institutional support. An initial set of requirements was presented to Phase 1 Office Lead, Frank Culbertson, on Aug. 25, 1995.

Within two weeks, the initial laptop configuration for the COSS was selected. A very significant factor in the selection of a delivery platform was the availability of some previously used shuttle flight certified units. Additional flight certification requirements above those required for shuttle were required by the Russian Space Agency (RSA), and were conducted by NASA contractor Lockheed-Martin. Lockheed-Martin was also responsible for integration of the flight equipment into the manifest flow, including acceptance testing by the RSA.

This development for the first delivery of the COSS software started on October 10, 1995, with the arrival of the development software, and was complete on January 31, 1996, to support NASA-Mir/Long Duration Mission 2 (LDM2). The completion date was firm due to the fact that the COSS platform and all of the support peripherals and software had to be stowed for transfer with the NASA astronaut on STS-76.

There are many key factors that enabled such a short development cycle for flight. The first is that the FDT project had developed conceptually much of the ground work, and already had some of the software and hardware ordered that was used to support the COSS project. The second is the significant ground work and progress of the initial Tiger Team. A third critical factor was the already in place NASA Shuttle (Code M) funding in support of FDT.

Although not an original part of the FDT project plan, the participation in the Phase 1 Program provided a tremendous opportunity. First, the FDT project will benefit from obtaining on orbit access to test the concepts in place for ISS. Secondly, in order to support Phase 1, the same logistics, configuration control management, and real time operations support required for ISS are necessary, only on a smaller scale. Since the COSS system and support structure is not currently designated as an operational requirement, changes to the methods and techniques can be made and tested without risk. This will allow the FDT system for ISS to be tested and operationally sound at the time it is integrated into ISS.

It should be noted that while the COSS platform is an example of a FDT, the COSS unit also provides psychological support, reference, and general office support in addition to training material. The psychological support includes commercial entertainment software, the astronaut's family scrapbook, and electronic books. The reference material available on the platform ranges from commercially available medical and geographical references to NASA developed, flight specific reference material. The office support software on the system contains a standard word processor, spread sheet, and daily planner software.

The development philosophy of the COSS team has been to look at how to reduce the costs of training, including delivering training at the appropriate time. As part of the charter for development, the team has placed emphasis on better real time crew support products, such as on-line reference material and task performance aids. In the future, procedures will be electronic, and augmented with additional levels of detail that a crewmember can access if more clarification is required. Currently, much training time is spent teaching astronauts when and under what conditions a standing procedure is valid, and when certain steps should be deleted or modified. As has happened with industry, the distinction between Just In Time Training and an Electronic Performance Support System (EPSS) is blurry, and may be a false distinction.

A large number of tasks were required to be completed by the FDT team to support the COSS project. These tasks included all software integration (COTS and non-COTS software), software configuration management, Flight certification test plan execution, and generation of CD-ROMs for flight, training, development and the ground support unit in Russia. Additional responsibilities included providing training to the LDM2 crewmember on interface and software applications, conversion and testing support for EDP/Hyperman, development of the Crew Worn Equipment and Emergency Egress/Escape refresher lessons, design and development of the Interface program to replace the Windows shell, and CD Stowage Location software generation.

The flight hard drives and flight backup hard drive were loaded with the final flight configuration approximately three weeks prior to launch. Post loading of the final flight hard drive, an error was found in the data files supplied by the Phase 1 Training group and used by EDP. Although the error was detected preflight, the COSS computer platform had already been integrated into the shuttle for launch. However, the EDP data file CD-ROM had not been generated, and a computer program was written and incorporated onto the CD-ROM in less than twenty-four hours. The patch program, when executed in-flight by the crewmember, fixed the EDP problem as well as provided minor updates to the interface. This technique, while not optimal, saved significant in-flight reconfiguration for the crewmember. Since the CD-ROMs are a late stow item, this method may be used for future flights.

There were many individuals from different organizations within the Mission Operations Directorate involved in this project. Their skills included programming, curriculum development, computer hardware and software, digital video generation, graphics, subject matter expertise and project management. Approximately twenty people contributed to the completion of the initial software delivery, however total labor costs were kept to less than one person year. The total funds spent by the Field Deployable Trainer project in support of the LDM2/COSS were approximately seventy-six thousand dollars for contractor labor, COTS software and development hardware.

The current COSS system has over one hundred software applications. A complete Flight Set of CD-ROMs consists of fifty-six CD-ROMs. Three complete sets and one partial flight-like set were generated for LDM2. The first complete set is the flight set, and is currently on board Mir. The second complete set is the Training set, and is currently in Russia and being actively used for training. The third complete set is for use in Russia by the Control Center, for real time support in the event a question or problem occurs. The partial set is located at Johnson Space Center, and is used by the FDT team for near real time support and development. Although the current standard for flight computers is to have a complete backup set of all software in bonded storage, this requirement was waived as a cost saving measure. A complete set costs approximately four thousand dollars.

COSS Hardware

The COSS platform currently on board Mir consists of a modified IBM Thinkpad 750c (33 MHz, 486 CPU) notebook PC based system with a docking station. The Dock I station contains a 2x CD-ROM drive and sound card which provides full multimedia capabilities when a headset is used. The Thinkpad has twelve Megabytes (MB) of Random Access Memory installed and is equipped with a three hundred forty MB removable hard disk, the largest available for the Thinkpad 750c. The COSS system also contains a video tuner that allows the computer's full screen to be used as a video monitor. As part of the video system, a Cannon 8mm camcorder model L1 is also included and when connected to the video tuner of the Thinkpad allows the user to view video tapes on the 9.5 inch monitor. The video tuner is installed in place of the 3.5 inch floppy disk drive.

It was decided the COSS system would not include a floppy disk drive. This was to help maintain configuration control and reduce the possibility of a computer virus infecting the system while on orbit. A spare hard drive was also flown, configured with the exact configuration of the prime flight hard drive.

The fifty-six CD-ROMs were stowed in six commercially available CD-ROM cases that were modified for flight. Each case is capable of holding twelve CD-ROMs. Commercial cases were chosen due to limited stowage onboard the Mir Space Station, as well as to keep costs as low as possible

Power for the COSS platform is supplied by Mir services. A power converter/adapter and associated power cables were built by the hardware contractor, Lockheed-Martin, and can be connected to any of the standard power outlets on board Mir as well as the shuttle orbiter. A converter was required, since the IBM Dock I station only accepts standard U.S. Alternating Current, which is not available on board Mir.

COSS Software

The software in use on the COSS unit can be divided into two broad categories: commercial off-the-shelf (COTS) and custom developed. For cost savings, COTS software was used whenever possible. A listing of the COTS applications is shown in Table 1, with the custom NASA developed software in Table 2.

The most noticeable piece of custom software in use on COSS is the Windows interface replacement. The custom interface replaces the program manager within Windows 3.1 and adds functionality while simplifying usage of the system. The interface was written using Borland's Delphi. Use of a compiled Rapid Application Development language allowed the Windows shell to be replaced. The Program Manager file was left on COSS and may be run from within the interface if necessary.

The custom interface uses a tabbed notebook metaphor and divides the programs into five categories: Applications, Training, Reference, Diversions, and Windows. The COTS software associated with each page is shown in Table 1, with the exception that the Windows page contains the standard Windows utilities (File Manager, Clock, etc.) along with the programs shown in Table 1 under Utility.

Along the bottom of the interface screen is a common area that has an exit button, a timer button, CD stowage location button, two bar graphs displaying the amount of free User and Graphical Device Interface resources, and the date/time for Moscow and Houston. The bar graphs showing free resources are for troubleshooting use only.

Applications	Reference	Diversions	Utility
Ascend	A.D.A.M. The Inside Story	Art & Music - Impressionism	Manifest
Lotus Organizer	Dr. Schueler's Home Medical Advisor Pro	Art & Music - Romanticism	Norton AntiVirus
Microsoft Excel	Encyclopedia of Science Fiction	Art & Music - Surrealism	Norton Utilities
Microsoft Word	Global Explorer	Art & Music - The Baroque	
	Library of the Future, 3rd Edition	Art & Music - The Eighteenth Century	
	Microsoft Encarta '95	Art & Music - The Medieval Era	
	Multimedia Family Bible	Art & Music - The Renaissance	
	Multimedia U.S. History	Art & Music - The Twentieth Century	
	Multimedia World Factbook	Beethoven and Beyond	
	Multimedia World History	Chess Master 4000 Turbo	
	Physicians Desk Reference	Echo Lake	
	The Dead Sea Scrolls Revealed	History of Music - American Folk Music	
	U.S. Geography: Alaska & Hawaii	History of Music - Music and Culture	
	U.S. Geography: The Land & Its People	History of Music - Romanticism to Contemporary	
	U.S. Geography: The Midwest	History of Music - Through the Classical Period	
	U.S. Geography: The Northeast	History Through Art - Ancient Greece	
	U.S. Geography: The Rockies	History Through Art - Ancient Rome	
	U.S. Geography: The Southeast	History Through Art - Romanticism	
	U.S. Geography: The Southwest	History Through Art - The Baroque	
	U.S. Geography: The West	History Through Art - The Enlightenment	
	Visions of Mars	History Through Art - The Middle Ages	
		History Through Art - The Pre-Modern Era	
		History Through Art - The Renaissance	
		Multimedia Mozart	
		Multimedia Schubert	
		Multimedia Strauss	
		Multimedia Stravinsky	
		Myst	
		Sharks! An Interactive Journey	

Table 1
COTS Software

Icons are used whenever possible to identify functions. A hint showing the full name of the program or function is displayed if the user pauses the cursor over an item. If the user wishes to learn more about that item they can click the right mouse button and an informational text window is displayed describing the program/feature.

The interface also assists the user by checking for the proper CD-ROM in the docking station drive prior to executing a selected program when that program requires a CD-ROM. If the drive is empty or contains the incorrect CD-ROM, the interface displays a window showing what CD-ROM needs to be installed and where that CD-ROM is stowed, along with what CD-ROM is currently installed (or none) and where it should be

General	Training	Reference
Windows interface replacement Restore software	Communication Specs/Displays Crew Worn Equipment Electrical Power System Malfunctions Electronic Documentation Project CBT Emergency Egress/Escape Environmental Controls and Life Support System Malfunctions General Purpose Computer (GPC) Replacement Guidance, Navigation and Control Orbiter Docking Systems Reaction Control System Redundancy Management	Electronic Documentation Project

Table 2
NASA Developed Custom Software

stowed when removed. This prevents any unexpected program faults and cryptic error messages from being displayed.

The timer function was added to the interface upon request of representatives of the Astronaut Office and includes up to four multifunction count-down timers, two stopwatches and twelve long-term alarms. All time functions include the ability to add reminder text to be displayed when the alarm executes.

The interface monitors all Windows programs and logs the start and end time associated with each. This monitoring occurs whether the program is executed via the interface, from the File Manager, or by first executing the Program Manager. The log will be returned with the hard drive at the completion of the mission for evaluation. Study of the log will assist the COSS software team in selecting programs for future missions. This capability is to be enhanced on future missions to include performance recording on selected software.

Two CD-ROM based refresher lessons were developed specifically for LDM2 and future Mir missions and include numerous firsts for astronaut training. These are the Crew Escape Systems and Crew Worn Equipment multimedia Computer Based Training (CBT) lessons. These lessons are the first designed to be taken both in flight and pre-flight, with the subject matter tailored to the situation. The student may also take the Crew Worn Equipment lesson in either English or Russian, another first in NASA training. The lessons utilize the full potential of the COSS platform and include video segments, sound and animation in conveying the training material. The lessons were developed based on LDM1 post-flight feedback.

The GPC Replacement lesson is designed as a Just-In-Time training tool. It was developed by the Ames Research Center in coordination with the Johnson Space Center and uses the Hypertext Markup Language. The lesson may be taken over the Internet, but in this case the web browser software is configured to use local files instead of remote files from a network.

The remaining lessons included on the COSS platform, the Communication Specs/Displays, Electrical Power System Malfunctions, Orbiter Docking Systems, Environmental Controls and Life Support System Malfunctions, Guidance, Navigation and Control, Reaction Control System Redundancy Management and the Electronic Documentation Project (EDP) CBT, are all traditional non-multimedia CBT lessons in use over the network at JSC. The lessons, with the exception of the EDP lesson, are Shuttle training lessons and included primarily for demonstration purposes. The EDP lesson provides training on a reference application used on COSS which is explained below.

The participation of the Electronic Documentation Project on LDM2 with the COSS team was primarily pursued to evaluate the on-orbit suitability of the EDP tools and techniques that have been previously developed and deployed for use within the Mission Control (MCC) facility.

The EDP viewer was developed specifically to meet the document utilization demands unique to MCC environment. These include rapid and non-linear traversal through a large library with heavy emphasis placed on both automatic and user-configurable hyperlinks and annotation capabilities. Additionally, since the system is designed to be portable across a variety of platforms, the use of the PC/laptop client by the COSS team is a natural extension of the capabilities of this software used on the Dec-Alpha workstations of the MCC. Early rehosting of EDP for the PC environment was accomplished specifically to support the COSS.

An additional goal of integrating EDP into the COSS, already successfully demonstrated, was that a large document library, over 700 pages in this case, could be assembled, checked out, and delivered in a short period of time for crew onboard use.

Based on this experience, the Space Shuttle Program will begin flying the EDP viewer as a portion of its standard portable computer load with STS-80, with the ultimate goal of reducing one half of a standard Flight Data File middeck locker in paper book storage. This will have the benefit of a recurring (each flight) savings of eight hundred thousand dollars.

From the Space Station perspective, a program requirement of hosting common applications onboard as are used in the control center will also be satisfied through use of the EDP viewer. The STS-76/Mir flight opportunity has proven instrumental in demonstrating that the EDP viewer, as an operations-oriented tool, is ideally suited for this task. Work has already begun with ISS engineering and operations teams to ensure that the library utilization and maintenance is handled in an optimum fashion.

Current experience has also shown that an entire compliment of Flight Data File can easily be placed on a single CD-ROM for use by a crewman with a laptop computer on-orbit. Continued development and optimization of the viewer for use on the LCD screen of a laptop is also planned.

The final custom software developed for COSS is a utility that we hope is never used on-orbit: the emergency restore utility. This floppy disk and CD-ROM combination is designed to restore functionality of the COSS unit with a minimum of data loss in the event of hardware or software problems. To use, the video tuner must be removed and the floppy drive from the Mir Integrated Payload System (MIPS) computer installed. The system can then boot from the restore floppy, which automatically runs the restore utility.

The restore utility program is a DOS program with Windows-like features, written in Borland Pascal. The main menu is cursor controlled and provides four options: perform a hard drive check and repair using Norton Disk Doctor (NDD), perform a file-by-file restore, perform a format of the hard drive followed by a file-by-file restore, or perform an image restore of the hard drive. The utility is able to restore the system with these options due to the fact that the CD-ROM provides sufficient storage capacity to hold two copies of the hard drive pre-launch configuration.

If a restore is required the first menu option, the hard drive check and repair using NDD, is the "safest" option for correcting problems while maintaining data integrity, in that only the damaged data is lost. The file-by-file restore should result in a working system without loss of data, however, any changes in user options in-flight would be lost. If the system is completely unusable the last two options will restore the system to a working state (with loss of new data) provided there is no catastrophic hardware failure. If the hard drive has no new bad clusters an image restore provides the safest restore method, and is the feature used to originally configure the flight hardware. The option of a format followed by a file-by-file restore has the advantage of providing a usable system even if the hard drive has developed bad areas during the flight. Future COSS systems will investigate methods of backing up user data on-orbit.

Human Spaceflight Firsts

There were many new features developed for this project which were firsts for the U.S. human space program. There was concern about the reliability of the system since the same platform was being used for the MIPS platform onboard Mir. Some of the problems have been attributed to unofficial Russian software being loaded onto the MIPS computer containing computer viruses. Another concern was possible hardware failure or a user reconfiguring the systems files such that the system would stop working.

The in-flight hard drive restore capability was developed in an effort to address these concerns and was a first in the program. The in-flight restore CD-ROM provides an added level of confidence that hard drive problems with the COSS unit can be dealt with on orbit if required To date the shuttle orbiter Payload General Support Computer and the MIPS platform does not have this capability. This ability to completely restore a laptop computer on-orbit provides a safety feature which will definitely be required on long duration missions.

While CD-ROMs computer systems have been tested on board the shuttle orbiter for potential operational use, this is the first time CD-ROMs have been used in a designated operational system. Since a number of the CD-ROMs were not commercially produced CD-ROMs, and were CD-Recordables, the zero-gravity performance was unknown. To date, no abnormal CD-

ROM performance has been noted. The flight of CD-Recordables is also believed to be a first.

The number of CD-ROMs flown to support this project is also a first in the Space program. There has never been any other flight that has flown such a large number of CD-ROMs, both commercial titles as well as NASA developed titles.

The operational use of training materials in flight has been conducted previously by the Russian space program. This is, however a first for the U.S. space program, and will provide valuable operational experience in preparation for ISS.

The development of a custom user interface for the COSS was a first for the Space program. The varying computer skill levels of the user community dictated the custom user interface described in the previous section. While customization of software for space flight for individual astronauts has been strongly discouraged by other application developers, the FDT team is encouraging the practice and will provide the service throughout the Phase 1 Program.

The development time for this system from ground zero to a flight certified unit (both hardware and software) was under six months. This is an extremely short cycle for flight systems, and unheard of for software applications development.

Future Plans for the FDT Project

Based on the user acceptance of the current COSS, additional applications and hardware upgrades are underway. The future deliveries of software and hardware will significantly increase the capabilities and potential for the COSS and OBT.

Planned hardware upgrades will include flight certification of a Pentium-class laptop and communication hardware. The upgraded laptop will not only increase performance, but allow for more applications to be based on the hard drive and allow for including software applications that have unacceptable performance on the current platform being used.

Software applications are currently planned that will provide more OBT and provide more real time support for the on board crewmember. More advanced data capture software, which will include skill and knowledge probes, is currently under development. Software for remote configuration management is also planned, and will work in conjunction with the communications hardware for remote on orbit software modification. The hardware and software combination will also allow the astronaut the capability to send and receive electronic mail while on orbit, as well as direct Internet access. The same system will be used for remote training support for crewmembers preflight.

Summary and Conclusions

Clearly, the potential for field deployable training is just now being realized. With the rapid increase in portable computer platform capabilities, the ability to provide high quality training in a remote setting is approaching that which would be found in a traditional training facility.

Technologies for instructorless training, such as Computer Based Training lessons and Instructorless Computer Aided Training are maturing, and will allow field deployable training to become a reality. When coupled with the various emerging virtual reality products, remote training can potentially provide a one for one replacement of the current training fidelity levels, while concomitantly reducing costs and increasing operations capabilities.

While all the technologies required are not yet mature, all are beyond the concept and early development stages. By the aggressive use of COTS products, NASA can also leverage against commercial investment to radically reduce the costs of training and operations support material development. As this project has demonstrated, robust in-flight products can be produced for extremely low cost, provided the correct project management methodologies are used.

Acknowledgments

The Crew On Orbit Support System and future Field Deployable Trainer could not be accomplished without the valuable support of many people: Charles Holliman, NASA Headquarters - Code M; Frank Culbertson; Charles Stegemoeller and Gary Kitmacher, Phase 1 Program Office; Robert Holkan and Barbara Pearson, Simulator Operations and Technology Division; Frank Hughes and Pamela Peters, Spaceflight Training Division; Anthony Griffith, Operations Division; Scott Gill and David Pogue, Systems Division; and Patrick Wilson, Avionics Systems Division.

AIAA-96-3502-CP

Operations Testing in a Concurrent Development Environment

Roger A. Burke
Simulator Operations and Technology Division
NASA Johnson Space Center
Houston, Texas 77058

Abstract

Real-time flight simulation is a major endeavor at Johnson Space Center (JSC) where flight crews and ground controllers receive the majority of their training. NASA is in the process of developing a large, complex simulator to support the training required for buildup of the International Space Station (ISS). This simulator, the Space Station Training Facility (SSTF), will have to be put into operation while still early in the development phase. This dictates an operations concept that is evolvable: tunable and adaptable to ever changing capabilities. Coupled with ever shrinking budgets and collapsing schedules this presents a formidable challenge to delivering a trainable simulator. To meet this challenge, NASA, along with the development contractor, has proposed a new testing philosophy: a "Test by Using" (TBU) archetype based on an Integrated Product Team (IPT) approach. This paper describes a concurrent Trainer Qualification Test (TQT) process designed not only to verify integrated system performance but also to provide operations validation early in the development cycle. Information will be provided on how a model of the proposed trainer operations processes was developed, how that model provided early feedback to the developers on operations concerns, and how that model is being used to track the development of and test the delivery of operations tools and capabilities. The paper will also provide insight into the techniques used to insure the trainer will have the capability to support the activities associated with each flight during the Station build-up period. It will address model fidelity issues, integration activities, as well as mission and operations capability testing. It will also touch upon flight procedures verification and how that function plays a role in the overall qualification process. Areas of prime responsibility for testing will be identified and discussed with respect to the Integrated Product Team approach. With this new testing methodology NASA hopes to cut months off the schedule template while still delivering a simulator and a trained and qualified operations support team capable of providing crew and controller training.

Introduction

Throughout the history of the space program - from Mercury though Gemini, Apollo, Skylab, and Shuttle - NASA has relied on large complex simulators that could simulate the entire mission from launch through landing. In the past these simulators have been built by a contractor specializing in the field of real-time simulation. Typically the development, installation and acceptance of the simulator would extend over a number of years. The contractor would implement a given set of requirements and specifications, deliver and install the simulator on site, and proceed into a six to nine month formal testing period composed of Site Integration Tests (SIT), Site Acceptance Tests (SAT), and Site Operations Tests (SOT). Basically, SOTs were designed to provide training for support personnel: usually a different contractor. Performing SOT at the tail-end of the development, just prior to release for training, left little time to implement, much less validate, operations processes that matched the capabilities and tools provided by the development effort.

Today NASA is engaged in developing another large and complex simulation facility the Space Station Training Facility (SSTF). Development of the SSTF, which began more than 10 years ago, has suffered through as many re-planning and downsizing activities as the Space Station Project itself. Today NASA finds itself with a half built simulator whose infrastructure was designed for a much more ambitious Station program, whose operations concept failed to change as downsizing progressed, and whose budget has been, and will continue to be, severely reduced.

Integrated Product Teams & Operational Capability

Early in FY 96 several events happened that prompted a series of reviews of the SSTF project: its organization, methodologies, requirements, and operations and testing. First, NASA underwent yet another reorganization: new people, less familiar with Space Station or the simulation facility were assigned to the project. Second, the contract was changed from a Level-of-Effort to Completion Form with its focus on documented completion criteria.

A number of troubling findings came out of these reviews. We found we had no way of measuring where we were with respect to supporting training for any one flight. The contractor had been implementing and installing "objects" without mapping the installed objects to operational capabilities - what flight activities could be performed. This did not mean the contractor was implementing incorrectly, it simply meant we could not measure real progress.

We also discovered that many of the level A requirements were untestable. They were at too high a level of abstraction to develop objective test from.

At this point it was decided to change the development philosophy from the traditional "develop to requirements and throw the results over the fence for the users to contend with" to an "implementation of operational capabilities" by an Integrated Product Team (IPT) where all interested parties, the developers, the customers, and the users, get involved at all levels of the development process and share the responsibility of either success or failure all the way from design through code inspections, unit testing, subsystem integration testing, and finally to trainer qualification testing..

It was recognized that two levels of IPT's were needed. One at the subsystem level to insure the correct level of model fidelity (training capability) and another at the facility level to insure the trainer could be operated, maintained, and sustained (operations capability). It was this second tier, the Ops IPT, that developed the present operations concept and implemented a method of testing that concept early in the development cycle.

Concurrent Operations

As late as early FY'96 an operations plan or process had never been developed for the SSTF. A concept of how operations might be performed had been generated but never mapped into a process. Since this concept of operations for the trainer was based on the old Freedom configuration, the Ops IPT decided to start from scratch and develop a new operations process.

In order to support assembly training for Station build-up the training facility had to support training operations before the simulator had been completely developed or installed. This concurrent operations plan, and its maturation, had to be in sync with the SSTF's incremental development plan - a plan based on a six month cycle for the delivery of new capabilities (Figure 1). Each increment delivered both subsystem capabilities and operations capabilities that supported multiple flights.

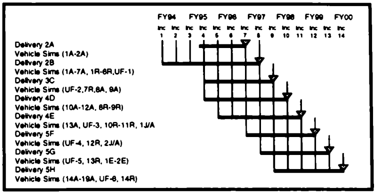


Figure 1 - Incremental Development Schedule

Knowing the facility had to support training crews and controllers while new equipment, crew station modules and software models were being installed and tested through the year 1999 prescribed an operations plan or process that had to be evolvable: it had to adapt as new training capabilities were added. Further, this ops plan had to be defensible in terms of recurring costs. In today's environment of continuous budget reductions, every dollar spent on operations must be justifiable or be in danger of being eliminated in the next round of cuts. To us these factors meant a small operations team with few external interfaces with other departments or organizations.

Operations Modeling

To be able to mature an operational plan and its associated capabilities incrementally, we felt it was necessary to begin testing the detailed plan at least two years prior to the delivery of the first training load - well in advance of having full operations capability. To do this it was necessary to: have a plan; map the plan to a set of processes that would execute the plan; develop a simulation of the processes; test the proposed processes using the simulation; and, finally, execute the operations processes during all phases of testing to validate the process. Of the above steps, we felt the development and execution of a simulation of the operations processes was critical to overcoming the lack of delivered operations capabilities.

This first phase of this effort was to develop a new operations concept with a primary goal of providing support to the users (i.e. instructors) with a minimum number of personnel while still providing the support required to keep these users satisfied with the product and services they were receiving. A secondary goal was to reduce communication time and errors by minimizing the number of interfaces between organizations and functional groups.

The initial concept was very simple - at least it looked simple on paper.

- *Provide support to those real-time activities that are directly associated with training. These support functions include:*
    - ⇒ *Morning readiness tests to insure all assets are operational.*
    - ⇒ *Coordination with external facilities to verify connectivity and correct data flow.*
    - ⇒ *Stowage support for laptops, headsets, and standard tool set.*
    - ⇒ *Configuring the assets for each training session (Configure for Training).*
    - ⇒ *Limited real-time failure recovery.*
- *Provide pre-session reconfiguration for:*
    - ⇒ *On-board short term plans*
    - ⇒ *Consumables*
    - ⇒ *State vector changes*
    - ⇒ *Electronic flight data file updates*
- *Corrective maintenance would be vendor supplied. LRU's will be sent out for repairs with call-in maintenance for those items that cannot be shipped. For critical failures, operation must be restored within twenty-four hours.*

Phase two was to map the concept to a simulation of the physical process using a process modeling tool. Actually two separate models were developed. The first model defined the daily operations the physical processes which really represent a cost, time, motion analysis of operations. The second model was a Customer Satisfaction and Perceived Quality (CS&PQ) model which attempted to identify the drivers that would determine how the users liked what they got.

Physical Processes Model

One advantage of using a modeling tool and trying to develop an actual simulation of a process is that it forces a detailed analysis of the process, its interfaces, and how they interplay. It turned out that the operations process turned out to be a set of smaller processes or tasks. The sequencing of these independent tasks define the actual operations process. Figure 2 shows a first cut at simulating a typical task. This task, called "Configure for Training" (CfT), is one of the simplest of ops scenarios taking only 15 minutes to perform and it interfaced, we thought, with only one other functional group - scheduling. This turned out to be a simplistic view. It did not consider where the configuration file came from - it is actually built a week or more in advance from information delivered to scheduling from the users - nor did it consider what is done if there was a change in plans or an error in the initial information. Figure 3 shows the same process after considering all these factors. The process complexity had grown to six interfaces between four separate organizations or departments. This growth in complexity was typical of the dozens of operations tasks that were modeled. When complete, the Physical Processes Model had 82 distinct communication interfaces between 18 separate organizations or departments in addition to 14 product handovers between various groups.

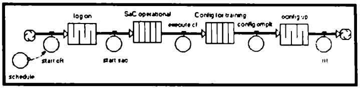


Figure 2 - Initial Configure for Training

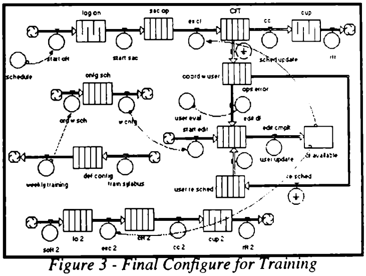


Figure 3 - Final Configure for Training

Execution of the physical process simulation indicated a number of problems with our original concept. First, to keep the number of personnel to a manageable level many of the processes had to be "interruptable" in the event a higher priority task, such as responding to a critical hardware failure, occurred. Second, that while on the average operations could be supported with a very small number of people, during peak activity periods the support needed increased by a factor of three or more even with interruptable processes. One criticism of the plan was that operations was sort of a "bang-bang" process: short periods of intense work followed by long periods of idleness. It was also found that under certain circumstances the users had to learn to perform the same processes as the ops personnel. As a result of these tests a number of tasks were reassigned to different organizations theoretically resulting in a 40% decrease in communication interfaces, a 50% decrease in product handovers, all of which resulted in a smoother personnel usage curve.

CS&PQ Model

In addition to the physical model we also wanted to understand the relationship between operations support and user satisfaction and with the user's perceived quality of the product. For nearly twenty years NASA has been operating and sustaining another large and complex training facility: the Shuttle Mission Simulator (SMS). For nearly all of these years, the SMS has provided high fidelity training day-after-day, week-after-week yet, despite a proven track record, it has gained a reputation as a fickle lady of dubious pedigree. Granted, the first few years of operation were a little shaky but certainly over the past ten years user utilization has hovered in the high 90 percentile and model fidelity has been proven time and time again by its ability to reproduce real-word signatures. Why then this "It's a lemon" mentality?

In the modeling tool's documentation was an example of human resource-based representations. Of particular interest to us was the circular relationships between morale, average knowledge base, team strength, and performance the model contained. This nest of circular relationships operate as feedback loops causing morale and performance to spiral either upward or downward. Many human resource situations are characterized by the operations of such feedback loops. The SMS's history has been punctuated with periods of roller-coaster ride plummets in user satisfaction and perceived quality followed by a gradual recovery only to plummet once again - a cycle that seemed to have about an eighteenth month period. It was to understand this cycle that led us to develop the CS&PQ model. The tool's example seemed to offer some explanation of the roller-coaster phenomena.

Using this model as a base, we added another inner loop element called *User Satisfaction* that affected both team strength and morale and was in turn driven by performance and morale plus a secondary loop called *Perceived Quality* that affected and was affected by *User Satisfaction*.

Determining the feedback effects between the various elements of the model was somewhat subjective. Although, for the past year, the users of the SMS have been grading their level of satisfaction against eighteen criteria, the data has almost no correlation. Despite this, drivers for *User Satisfaction* and *Perceived Quality* were based on this sketchy recorded data plus a lot of "gut feel" from experienced operations personnel.

Our experiential knowledge base indicated that the primary drivers of *User Satisfaction* were: the user interface - is it easy to do the job, and lost time - the time spent waiting. Both of these drivers are drainers - they don't increase satisfaction they only tend to decrease it.

These days impressive user interfaces are an integral part of a workers office environment. They expect no less sophistication in their trainers. We used mouse clicks as one performance measure with five being the norm for executing any single task. Another performance measure was the intuitiveness of the display and sub-displays. Our measure here was how many times a user had to refer to the users manual to accomplish his daily tasks.

Lost time, whether from a sim crash or from other delays, is fairly self explanatory and the performance measure - how much time does the user spend sitting around waiting for the simulator to be ready - quite straight forward.

There is one other drain on user satisfaction - spontaneous decay. Left to its own doing, with no active stimuli to sustain it, *User Satisfaction* will dissipate. Just how fast it will dissipate is arguable but it will decline. It's like buying a new car. At first it's fun to drive but, after the newness wears off, it's not so enjoyable any more. Little things that, at first, went unnoticed begin to annoy you - a squeak here, a rattle there. A trip to the dealer to fix these little annoyances gives some relief with a return of some of the original enjoyment but even this quickly dissipates. User satisfaction with a simulator follows these same rules: no matter how "good" the simulator nor how high the quality after the honeymoon is over, and with no new capabilities added, the users will become dissatisfied.

In our model, the drivers of the *Perceived Quality* element were two related factors: the rate of error discovery and the total number of perceived errors. Both of these are really a measure of the capabilities delivered. Perceived quality is inversely related to both rate of discovery and total errors. We refer to the total number as perceived errors because these represent a backlog of work that needs to be done and, as with any backlog, there are duplications of problems reported. There can also be a substantial number of non-problems reported as problems. In any respect, a long list of unresolved perceived problems will almost guarantee a low quality rating. For example: at one time on the SMS, when user satisfaction and perceived quality were at one of the periodic lows, there was a large backlog of problems - well over 2000. To reduce this backlog to a manageable number a simple solution was employed: any problem over 90 days old was simply thrown away. Amazingly enough, even though no technical improvements were made to the simulator, the perceived quality shot up with user satisfaction following right along. Appearances count!

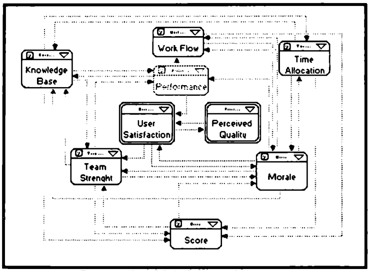


Figure 4 - CS&QP Architecture

We added an additional nested loop in the *Perceived Quality* element. As perceived quality decreases the error discovery rate increases somewhat. As the users become more and more disenchanted with the simulator they tend to nit-pick; reporting minor problems that they overlooked before. This turns out to be important in our situation since the operations concept had planned to fix only critical problems rapidly, leaving minor problems to be fixed in some subsequent increment delivery. In other words, we fully expected out problem count to increase over the life of a particular training load release. Figure 4 represents the architecture of the CS&QP model showing the basic elements and their interfaces.

Simulating the *CS&PQ* model provided some rather disheartening results. We could certainly reproduce the so called "death spiral" (Figure 5) so often seen in the SMS but not the recovery. Tuning the interactive factors only changed the rate at which the model spiraled downward but not the basic signature. The overall controlling element seemed to be *User Satisfaction* and in particular the spontaneous decay function. After the honeymoon period, if there is no input to sustain or increase satisfaction, *User Satisfaction*, *Morale*, and *Perceived Quality* feed each other downward. Our basic model had no sustaining input. Our operations concept of postponing fixes for non-critical problems, left us with no bolstering stimuli specifically, something new in the training load to periodically re-excite the user. The situation is analogous to taking your new car back to the dealership to fix that annoying squeak or rattle. If the dear fixes the problem your spirits are buoyed up: sort of a second honeymoon if you will. If not, you become more irritated and your satisfaction with the automobile suffers as a consequence.

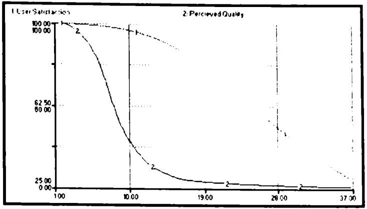


Figure 5 - User Satisfaction Death Spiral

To stabilize the *User Satisfaction* element of the model we modified the operations concept to allow fixes for non-critical problems to be fed back into the existing training load. This helped the *Perceived Quality* element since the backlog of problems decreased but more importantly, as modeled, it acted as a "shot-in-the-arm" for the *User Satisfaction* element. Again, this was based on observations made on the SMS over the years. These additions were enough to sustain both user satisfaction and perceived quality.

Trainer Qualification Testing

Trainer Qualification Test (TQT) was intended as a replacement for what was historically Site Acceptance Tests (SAT) and Site Operations Tests (SOT). Other testing activities, in addition to TQT's, were in the basic development template. In all, some

24 to 30 weeks of simulator testing for each flight were being planned (Figure 6). The objective of the Trainer Qualification Tests was to verify the Level A requirements to satisfy contractual stipulations. Unfortunately, due to the incremental and concurrent development approach required for this project, the majority of Level A's would not be fully satisfied until the development period was complete - sometime in FY'99. Thus we were left with essentially nothing to test until FY'99: two years after training had started. In mid FY'96, after several attempts at revising the TQT philosophy, we took a close look at the entire testing sequence - what we were doing, who the participants were, and what we were trying to prove.

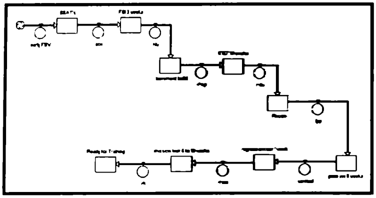


Figure 6 - Original Testing Template

The original testing template was divided into three levels of testing each with its own configuration and load management protocol. Level 1 was aimed at model fidelity testing and consisted of Subsystem Acceptance Test (SSAT) and Flight Software Integration test (FSI). Level 2 was called Increment Trainer Qualification Test (ITQT) and was designed to validate compliance with the level A requirements. Both Level 1 and Level 2 testing was to be conducted by the developers on a non-flight specific basis.

Level 3 was flight specific testing consisting of Regression Test - conducted by the developers, Procedure Verification - conducted by the users, and several periods of Mission Tests - conducted jointly by the users and developers.

Since this string of testing functions represented a process, we chose to model the sequence using the same process modeling tool that had been used for modeling the operations processes. This model included the additional processes required to fix a problem once it was found (why test except to find and fix errors?). In developing the model a couple of facts became obvious. All testing activities required a core set of simulator capabilities such as moding; data logging, archival, and reduction; and initial condition generation. All testing, with the exception of SSAT's and possibly regression tests, required flight software and in fact all these test were doing about the same thing for essentially the same reason: to determine if the simulator would support the activities that were planned for the particular flight. Finally, the identical process was required to identify and fix a problem found during any of the testing requiring flight software. The realization that all these tests were essentially identical led us to propose a completely redefined and streamlined testing philosophy (Figure 7).

The new concept calls for Instigating a "Test by Using" (TBU) philosophy for that core set of simulator capabilities required by everyone. This core set of capabilities is normally referred to as the infrastructure. "Test by Using" simply means that a dedicated set of tests will not be run for infrastructure functions or capabilities. Since these functions are required to work in order to successfully complete the real testing they will be tested by default. Any function that does not get tested this way probably isn't needed anyway.

The proposed process collapses the present FSI, ITQT, Procedure Verification, and Mission testing functions into a single mission specific Trainer Qualification Test function. This makes sense for a couple of reasons. First, according to our analysis there was about 70% overlap in testing coverage. Second, the same set of Subject Matter Experts (SME's) had to be present during the testing to evaluate the simulator performance and, most importantly, to determine whether it anomaly is a flight software problem or a simulator problem. This was true for all the different testing phases once the SSAT's had been completed. The same SME's would be involved in trouble shooting, and fixing the problems and any subsequent re-testing required. To be quite blunt, only a small number of people are going to be capable of doing the testing effectively so our concept was to build the team and do the testing once.

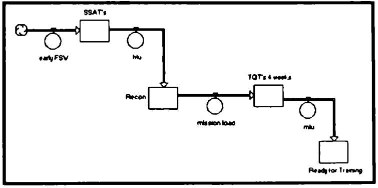


Figure 7 - Proposed Testing Process

An important part of the plan was to execute the TQT's in an operational (i.e. training) environment. Conduct this testing just like it was a training session. By assigning operations personnel, instructors, virtual crewmen and controllers to do what they would do during an actual training session using the tools available then we will test not only the simulator capabilities but also the support processes and personnel necessary to conduct a training session. Most importantly, we want to build a team that is conditioned to working in an operational environment and to be able to quickly answer or investigate problems or questions that arise during actual training. This may sound like an obvious step but historically, users, trainee's, and operations personnel had been relegated to observers rather than participants of the testing process with the testing conducted in a sterile, controlled environment. As a result, initial performance by the operations team has always been a little shaky.

The elimination of regression testing was also a piece of the proposal. Historically regression testing has proven to be a waste of time and resources. The criticality of the problems that were found did not warrant the effort of finding them particularly since most were never fixed because they was such a nit. It is much more cost effective to find these types of problems during training operations - if someone complains then it's probably worth fixing.

The key ingredient to making this proposed process work is the development of the tools necessary to isolate the cause of a problem uncovered during the testing phase. It is imperative that we be able to distinguish between flight software problems and simulator problems. Flight software issues need to be fed back to the Station program in a timely manner in order to get resolved prior to actual flight. Simulator problems need to be fixed quickly by the simulator developers, a completely separate organization, in order to support training for the flight. Experience with the Shuttle Mission Simulator has shown that without proper tools, the determination between flight software problems and simulator problems took one to two weeks. That was not the time it took to fix the problem, only to identify who's problem it was. Today, with the development of a number of evaluation tools, this time has dropped to one to two days.

One controversial element of the proposed testing plan was to eliminate all off-nominal testing (i.e. malfunctions) and concentrate only on verifying nominal capabilities. This greatly reduces the test matrix required to qualify the trainer. It does however, leave a large segment of the training scenarios unqualified. Three factors mitigate this risk. First, malfunctions are tested during Sub-System Acceptance Testing. The model's failure signatures are verified. This represents some level of confidence that they will work in an integrated environment. Second, experience has shown that unexpected system behavior, whether from a malfunction that didn't work properly or from some other source such as a hardware failure, often end up being good training cases. Just the act of chasing down some unknown signature to its root cause is a valuable learning experience. Crew and flight controllers have been rather understanding about being exposed to these situations. Once! Which brings us to the third mitigating factor: our experienced testing team, conditioned to working in an operational environment, should be able to fix the problem quickly particularly after the crews and controllers have, if not identified the source, at least eliminated many of the possibilities.

Summary

We feel the development of an operations simulation for the physical processes turned out to be very beneficial. It forced our thinking down to the level of detail required to discover just how complex a simple concept can be. It pointed out common processes used by multiple organizations which helped restructure assignments and product flows to simplify the entire plan. This simulation gives us a ruler from which to measure the actual performance of the operations team.

Physical processes are just a mechanism. It is people that bring processes to life and give them character. Our Customer Satisfaction and Perceived Quality model was an attempt to inject this human aspect. Unfortunately we have little faith in our CS&QP model as it stands. Even though it did tend to predict the Increment 5 fiasco, we feel more work is needed to understand to interplay between the various elements.

To date none of the proposed changes to the testing process has been implemented nor have either of the operations process models been validated against, nor tuned to, actual operations. An attempt was made during Increment 5 Qualification Testing to try executing the tests in a virtual operations environment with testing personnel acting as ops personnel and as users. Unfortunately, Increment 5 was not far enough along the development road to have anything operational except for some ops tools and displays that were close to final form and capability. We approached the testing of these functions as if we were inexperienced users - we read the users guides then tried to perform a set of tasks on line using these tools and displays. The result was disaster! By the end of the first testing session we were utterly frustrated and dissatisfied-satisfied. The user interface was so "unfriendly" and the users guides so poorly written that we gave up in utter frustration. In addition, the load was very unstable: staying up for only short periods of time and crashing for what appeared to be no known reason. Surprisingly, given the state of the interface and stability, the CS&PO model predicted just such a user reaction. For this and other reasons, Increment 5 Qualification Testing was terminated early without verifying a single requirement but it did point out that load stability had to be dealt with prior to the next increment release.

AIAA-96-3503-CP

SIMULATION EVALUATION OF A NEURAL-BASED FLIGHT CONTROLLER

Joseph J. Totah
Aerospace Engineer
NASA Ames Research Center
Moffett Field, California

Abstract

The objective of this paper is to present results from the simulation evaluation of a direct adaptive tracking controller. The control architecture employs a single pre-trained neural network to represent the non-linear aircraft aerodynamics in the model inversion portion of the controller. The aircraft model used for this evaluation is representative of the F-15 Advanced Control Technology for Integrated VEhicles (ACTIVE) aircraft. The controller was evaluated for two cases: (1) no knowledge of modified aircraft aerodynamics in the model inversion portion of the controller, and (2) complete, accurate, and instantaneous knowledge of the modified aerodynamics. The results were compared with the conventional mode controller in all cases. The results indicate extremely desirable airframe stabilization characteristics for case (2) that do not degrade significantly for case (1) as does the conventional controller. It was concluded that this controller exhibits both stable and robust adaptive characteristics when subjected to mild and extreme changes in the aircraft aerodynamics. Integration of an on-line neural network is planned to complete the implementation of this control architecture.

Introduction

In 1994 NASA awarded an Advanced Concepts Program Research Project to McDonnell Douglas Aerospace (MDA), entitled Intelligent Aircraft Control System. The objective of this project is to demonstrate a flight control concept that can identify aircraft stability and control characteristics using neural networks, and then utilize this information to optimize aircraft performance in nominal, off-nominal, and simulated failure conditions. The principle participants in this project are MDA, NASA Ames, and NASA Dryden Flight Research Facility (DFRF), with the F-15 ACTIVE aircraft shown in figure 1 to be used for flight demonstration at DFRF.

The project is divided into three phases. Phase I focuses on the development and flight evaluation of pre-trained neural networks for mapping the very large aerodynamic stability and control derivative database that currently exists for the F-15 ACTIVE aircraft's six-degree-of-freedom simulation. Phase II will be to employ advanced self-learning, or on-line learning methods to either update the stability and control derivatives or train on estimates of the total aerodynamic coefficients in nominal, off-nominal, and simulated failure conditions. This will culminate in a Phase III flight demonstration of reconfigurable control systems that will rely on both the pre-trained and advanced, on-line neural networks to provide the best available estimate of the aircraft dynamics.

The approach described above is similar to other efforts[1,2,3,4] with the exception that this program will perform actual flight testing in all three phases of the project. The direct adaptive tracking control architecture developed by Kim and Calise[1] was implemented by NASA for the on-line neural network development portion of the Phase II effort, and is currently being evaluated against other candidate controllers for the Phase III effort, as well.

The results presented in this paper treat two cases: (1) no knowledge of modified aircraft aerodynamics in the model inversion portion of the controller, and (2) complete, accurate, and instantaneous knowledge of the modified aerodynamics. The results represent a necessary first step to ultimately developing an on-line neural network that is optimized based on learning rate, size, accuracy, and the ability to generalize when training on local estimates of nominal or modified aircraft dynamics. These results are also considered a preliminary step to evaluating controller performance in a high fidelity piloted simulation environment prior to qualifying it for flight. Candidate on-line networks are currently being developed, and upon completion they will be integrated into this control architecture for evaluation relative to the results presented for case (1) and case (2).

F-15 ACTIVE Simulation

An F-15 ACTIVE simulation was developed at NASA Ames, and it is modeled after configuration G of the U.S. Air Force's Short takeoff and Landing Maneuver Technology Demonstrator (S/MTD)

Senior Member

program.[5] The model assumes gear and flaps retracted with no thrust vectoring capability. The conventional mode controller was modeled in addition to the direct adaptive tracking controller to drive the ailerons, canards (differential and symmetric), stabilator (differential and symmetric), and rudder. A simple first order dynamic system was assumed for both manual and auto throttle control, however they are not exact representations of the actual throttle logic.

The equations of motion are based on perturbation theory, however most of the non-linear terms were retained in the gravitational and inertial portions of the translational axes. All of the non-linear terms were retained in the inertial portions of the rotational axes.

A block diagram of the simulation is shown in figure 2, where NN1 is the pre-trained neural network used in the model inversion portion of the direct adaptive tracking controller's command augmentation system (CAS). The CAS and NN1 will be described in more detail in the next section.

Environmental conditions were also modeled to account for pressure, density, temperature, and speed-of-sound variations with altitude. This allowed for scheduling the aerodynamic derivatives, conventional control system gains, and nominal operating conditions throughout the flight envelope.

The model was integrated with a commercial graphics database, a simulator pod with simple auditory feedback, a three-axis hand controller, throttle lever, and push-button switches to provide an excellent, low-cost fixed-base simulator for pilot-in-the-loop simulation, shown in figure 3.

Model validation was obtained by matching perturbed state time histories at various flight conditions to those generated by the full, non-linear, six-degree-of-freedom simulation conducted by the Air Force during the S/MTD program.[5]

Direct Adaptive Tracking Controller

Controller

The direct adaptive tracking controller developed by Kim and Calise was implemented as designed, except for a modified model inversion approach that will be discussed later in this section. The controller is comprised of a command augmentation system (CAS) that has both an attitude orientation system and command augmentation logic. The attitude orientation system is an airframe stabilization system that accepts rate commands. This serves the function of stabilizing the airframe while following the commanded rates. The command augmentation logic is an outer loop function to track the pilot commands. The pilot commands normal acceleration with longitudinal stick, lateral acceleration with pedals, and roll rate with lateral stick. This design allows for normal acceleration and roll rate command tracking with automatic turn coordination.

The output of the attitude orientation system is transformed into commanded aircraft body axis rotational accelerations, which are then used in an .. inverted model to calculate control surface deflections. A flowchart for the CAS is shown in figure 4. The control laws defining the pilot input to control surface deflection and the design philosophy used therein is described in detail by Kim.[6]

The method developed by Kim and Calise to perform the inversion in T4 was modified such that a single neural network was used to model stability derivatives rather than total aerodynamic coefficients. T4 was only modified for the purpose of this study and subsequent comparison of this architecture with other controllers requiring a neural network model of the aerodynamic stability derivatives. Nonetheless, because the mapping of the stability derivatives was so accurate, the neural network was essentially an exact representation of the aircraft aerodynamics, and thus the T4 inversion was considered to be an exact inversion. The equations comprising T1, T2, T3, and T4 are given in the appendix along with the equations of motion. The aerodynamic stability derivatives in T1 and T4 are denoted with an overstrike, such as $\overline{C}_{Z\alpha}$, to signify that they are neural network estimates.

Although the stability derivative approach is sufficient for the objective of this paper, it may not be used for the Phase III portion of the program because it is acknowledged that in-flight estimates of total aerodynamic coefficients from aircraft sensors are, in general, easier to derive than estimates of individual stability derivatives.

Additional modifications were made to T4 to account for symmetric canard position, which is scheduled with angle-of-attack (the same schedule as that of the conventional mode controller) and ailerons and differential canard position, which are blended as:

$$\delta_a = 2\delta_t \quad (1)$$

$$\delta_{dc} = -\delta_r \quad (2)$$

The symmetric canard effect appears in A.33. The aileron effects appear in A.23 and A.29. Finally, the differential canard effects appear in A.25 and A.31.

Neural Network (NN1)

A pre-trained multi-layer perceptron neural network with a hybrid Levenberg-Marquardt (LM) solution technique was employed to model the aerodynamic derivatives used in the inverted model. The network structure consists of two inputs (M,h), 20 processing elements in a single hidden layer, and 32 outputs corresponding to each of the aerodynamic derivatives. The entire aerodynamic database was used for both training and testing, which ranged from 0.3<M<1.2 at sea level, to 0.6<M<2.0 at 50,000 feet. The data were scaled for training by subtracting the mean and dividing by the standard deviation of each derivative individually. Figure 5 shows an error histogram of C_{Z_α}, which performed within an acceptable error tolerance of $\pm 3.5\%$. Error tolerances were previously established for all aerodynamic derivatives based on earlier work performed at NASA.[7]

The LM results are considered outstanding. This is dramatically illustrated in figure 5 in which the LM results are compared with a variation of back-propagation called Quickprop (where the learning rate is automatically set for fast and accurate convergence). The Quickprop structure consists of two inputs (M,h), 20 processing elements in a single hidden layer, and 1 output corresponding to C_{Z_α}. It can be seen that the LM approach outperformed Quickprop by a factor of two with respect to error. A correlation coefficient of the desired and actual output across the epoch was the objective function used to optimize the Quickprop results when varying the number of processing elements in the hidden layer and the momentum term. The correlation coefficient is a function of the desired output, d, actual output, o, and the epoch size, E:

$$\sigma = \frac{\sum (d_i - d)(o_i - o)}{\sqrt{\sum (d_i - d)^2 (o_i - o)^2}} \qquad (3)$$

$$d = \frac{\sum d_i}{E} \qquad (4)$$

$$o = \frac{\sum o_i}{E} \qquad (5)$$

As a point of reference, the average LM correlation coefficient for all stability derivatives was 0.9947, with a minimum value of 0.9750. A comparison of this, and other performance metrics for LM relative to linearly interpolated data table look-ups (as implemented in MATLAB on a Silicon Graphics workstation) is provided in table 1.

Table 1. Performance of LM vs. Table Look-Ups

MATLAB Benchmark	Size (bytes)	Speed (sec)	Avg. Correlation
Table Look-Ups	33,341	0.4	1
LM	10,508	0.04	0.9947
Change	**3:1**	**10:1**	**-0.53%**

Results

The maneuver chosen to illustrate the CAS performance was the same maneuver considered by Kim and Calise for an F-18 aircraft. Starting at Mo=0.6 and ho=10,000 ft, a roll rate command of 10 deg/sec is given until an 80 deg bank angle is obtained. The normal acceleration is maintained at 1.0g during the roll maneuver. This is followed by a 5.0g normal acceleration command that is maintained until the end of the maneuver. The throttle is held at 100%, resulting in a speed of M=1.1 after 30 seconds.

First, a baseline set of responses are presented under nominal conditions. Next, two cases are treated: (1) no knowledge of modified aircraft aerodynamics in the model inversion portion of the controller, and (2) complete, accurate, and instantaneous knowledge of the modified aerodynamics. Both mild and extreme changes in the modified aerodynamics are considered.

Figure 6 depicts normal acceleration and roll rate response for this maneuver. In order to establish baseline performance under nominal conditions, the responses presented are for the conventional controller and CAS without the introduction of any modified aerodynamics. As shown in figure 6, there are differences between the two controllers. The most noticeable are the overshoot of roll rate response for the CAS, and the overshoot and time constant of normal acceleration for the conventional controller.

Figure 7 depicts controller responses to the same stick inputs for the same maneuver. However, modified aerodynamics are introduced that reduce the effectiveness of all control surfaces by 30%. Essentially, all of the aerodynamic control derivatives used in the equations of motion are multiplied by a factor of 0.7. The conventional controller normal acceleration response has noticeably more overshoot than shown in figure 6, and a roll rate response of approximately 8 deg/sec. On the other hand, the CAS performs quite well without knowledge of the modified dynamics.

Figure 8 depicts the same responses, except now the CAS has knowledge of the modified aerodynamics. In other words, the aerodynamic control derivative values estimated by NN1 are multiplied by a factor of 0.7 as well. The CAS normal acceleration and roll rate responses shown in figure 8 are nearly identical to those shown in figures 6 and 7.

In order to explore this in greater detail, an extreme condition was examined. Modified aerodynamics are now introduced that reduce the effectiveness of all control surfaces by 80%. For this case, all of the aerodynamic control derivatives used in the equations of motion are now multiplied by a factor of 0.2. The conventional controller normal acceleration response shown in figure 9 cannot maintain the commanded value at all, and the roll rate response can only achieve a value of 5 deg/sec for a short period of time. The CAS, however, again performs quite well without knowledge of the modified dynamics. It shows only a slight "wrinkle" in the normal acceleration and roll rate responses just prior to attaining the commanded value.

Figure 10 depicts the same responses, except now the CAS has knowledge of the modified aerodynamics.

Again, this means that the aerodynamic control derivative values estimated by NN1 are multiplied by a factor of 0.2. The CAS normal acceleration and roll rate responses shown in figure 10 are nearly identical to those shown in figures 6 and 7.

These results can best be explained by figure 11. Differential stabilator position is plotted for both mild and extreme conditions for both case (1) and case (2). The first thing to note is the shape of the differential stabilator response. The CAS tends to shape the control surface response whereas the conventional controller does not. This becomes more apparent for the extreme condition, whether the failure is known or unknown to the CAS. It is felt that this shaping is the attempt of the CAS to provide the exact amount of control surface deflection required to make-up for the reduced effectiveness and track the commanded input to achieve the desired response, whether NN1 knows about the reduced effectiveness or not.

Further, the CAS inner loop and outer loop time constants are set at 0.8 sec and 2.4 sec, respectively. This, perhaps, creates a high gain system, which might explain why the CAS performs well for both case (1) and case (2). The sharp jumps in control surface positon at the onset and completion of the control input tend to support this possible explanation.

Conclusion

This objective of this paper is to present the results of a simulation evaluation a direct adaptive tracking controller. The control architecture employed a single pre-trained neural network to represent the non-linear aircraft aerodynamics in the model inversion portion of the controller. The results indicate that the controller performs quite well for the two cases considered: (1) no knowledge of modified aircraft aerodynamics in the model inversion portion of the controller, and (2) complete, accurate, and instantaneous knowledge of the modified aerodynamics. The results indicate extremely desirable airframe stabilization characteristics for case (2) that do not degrade significantly for case (1) as does the conventional controller. It is concluded that this controller exhibits both stable and robust adaptive characteristics when subjected to mild and extreme changes in the aircraft aerodynamics. As a final note, other types of failures have been examined and they corroborate the results shown for the two cases presented in this paper.

Future plans are to integrate an on-line neural network into the direct adaptive tracking architecture. The controller will also be modified back to its original design to allow for mapping coefficients rather than stability derivatives. Finally, the controller will be evaluated in a high fidelity simulation environment for comparison with other candidate controllers currently being examined as part of the Phase III portion of this program.

Appendix

T1

$$P_c = \delta_{la} - p \quad (A.1)$$

$$Q_c = \left(\delta_{ln} - N_{Zcg}\right)\frac{g\left(K_1 + \frac{K_2}{S}\right)}{\left(U_0 + u\right)} - q \quad (A.2)$$

$$R_c = \left(\delta_{pd} - N_{Ycg}\right)\frac{g\left(K_3 + \frac{K_4}{S}\right)}{\left(U_0 + u\right)} - r \quad (A.3)$$

$$K_1 = t_n K_2 \quad (A.4)$$

$$K_3 = t_y K_4 \quad (A.5)$$

$$t_n = \frac{-mV_t}{\bar{q}S\bar{C}_{Z_u}} \quad (A.6)$$

$$t_y = \frac{mV_t}{\bar{q}S\left|\bar{C}_{Y_\beta}\right|} \quad (A.7)$$

$$K_2 = K_4 = \frac{4.6}{t_{so}} \quad (A.8)$$

$$t_{so} = 2.4\text{ sec} \quad (A.9)$$

$$\dot{\Phi}_c = P_c + \left(Q_c \sin\Phi + R_c \cos\Phi\right)\tan\Theta \quad (A.10)$$

$$\dot{\Theta}_c = Q_c \cos\Phi - R_c \sin\Phi \quad (A.11)$$

$$\dot{\Psi}_c = \frac{Q_c \sin\Phi + R_c \cos\Phi}{\tan\Theta} \quad (A.12)$$

T2

$$U_1 = K_{p\Phi}\left(\frac{\dot{\Phi}_c}{S} - \Phi\right) + K_{d\Phi}\left(\dot{\Phi}_c - \dot{\Phi}\right) \quad (A.13)$$

$$U_2 = K_{p\Theta}\left(\frac{\dot{\Theta}_c}{S} - \Theta\right) + K_{d\Theta}\left(\dot{\Theta}_c - \dot{\Theta}\right) \quad (A.14)$$

$$U_3 = K_{p\Psi}\left(\frac{\dot{\Psi}_c}{S} - \Psi\right) + K_{d\Psi}\left(\dot{\Psi}_c - \dot{\Psi}\right) \quad (A.15)$$

$$K_{p\Phi} = K_{p\Theta} = K_{p\Psi} = \frac{9.2}{t_{st}} \quad (A.16)$$

$$t_{st} = 0.8\text{ sec} \quad (A.17)$$

$$K_{d\Phi} = K_{d\Theta} = K_{d\Psi} = \frac{K_{p\Phi}^{\,2}}{2} \quad (A.18)$$

T3

$$\dot{P}_c = U_1 - U_3 \sin\Phi - \dot{\Psi}\dot{\Theta}\cos\Phi \quad (A.19)$$

$$\dot{Q}_c = U_2 \cos\Phi - \dot{\Theta}\dot{\Phi}\sin\Phi + U_3 \sin\Phi\cos\Theta$$

$$+\dot{\Psi}\dot{\Phi}\cos\Phi\cos\Theta - \dot{\Psi}\dot{\Theta}\sin\Phi\sin\Theta \quad (A.20)$$

$$\dot{R}_c = U_3 \cos\Phi\cos\Theta - U_2 \sin\Phi - \dot{\Theta}\dot{\Phi}\cos\Phi$$

$$-\dot{\Psi}\dot{\Phi}\sin\Phi\cos\Theta - \dot{\Psi}\dot{\Theta}\cos\Phi\sin\Theta \quad (A.21)$$

T4

$$\begin{bmatrix} \delta_t \\ \delta_s \\ \delta_r \end{bmatrix} = \begin{bmatrix} b_1 & b_2 & b_3 \\ b_4 & b_5 & b_6 \\ b_7 & b_8 & b_9 \end{bmatrix}^{-1} \begin{bmatrix} \dot{P}_c - L_1 \\ \dot{Q}_c - M_1 \\ \dot{R}_c - N_1 \end{bmatrix} \quad (A.22)$$

$$b_1 = \frac{I_{xz}\left(\overline{C}_{n_{\delta_t}} + 2\overline{C}_{n_{\delta_a}}\right) + I_z\left(\overline{C}_{l_{\delta_t}} + 2\overline{C}_{l_{\delta_a}}\right)}{\dfrac{I_x I_z - I_{xz}^{\ 2}}{\overline{q}Sb}} \quad (A.23)$$

$$b_2 = 0 \quad (A.24)$$

$$b_3 = \frac{I_{xz}\left(\overline{C}_{n_{\delta_r}} - \overline{C}_{n_{\delta_{dc}}}\right) + I_z\left(\overline{C}_{l_{\delta_r}} - \overline{C}_{l_{\delta_{dc}}}\right)}{\dfrac{I_x I_z - I_{xz}^{\ 2}}{\overline{q}Sb}} \quad (A.25)$$

$$b_4 = 0 \quad (A.26)$$

$$b_5 = \frac{\overline{C}_{m_{\delta_s}} \overline{q}S\overline{c}}{I_y} \quad (A.27)$$

$$b_6 = 0 \quad (A.28)$$

$$b_7 = \frac{I_{xz}\left(\overline{C}_{l_{\delta_t}} + 2\overline{C}_{l_{\delta_a}}\right) + I_x\left(\overline{C}_{n_{\delta_t}} + 2\overline{C}_{n_{\delta_a}}\right)}{\dfrac{I_x I_z - I_{xz}^{\ 2}}{\overline{q}Sb}} \quad (A.29)$$

$$b_8 = 0 \quad (A.30)$$

$$b_9 = \frac{I_{xz}\left(\overline{C}_{l_{\delta_r}} - \overline{C}_{l_{\delta_{dc}}}\right) + I_x\left(\overline{C}_{n_{\delta_r}} - \overline{C}_{n_{\delta_{dc}}}\right)}{\dfrac{I_x I_z - I_{xz}^{\ 2}}{\overline{q}Sb}} \quad (A.31)$$

$$L_1 = \frac{1}{I_x I_z - I_{xz}^{\ 2}} \Bigg\{ \begin{bmatrix} pqI_{xz}\left(I_x - I_y + I_z\right) - \\ qr\left(I_z^{\ 2} - I_y I_z + I_{xz}^{\ 2}\right) \end{bmatrix} +$$

$$\overline{q}SbI_{xz}\left[\overline{C}_{n_\beta}\beta + \frac{b}{2(U_0 + u)}\left(\overline{C}_{n_p} p + \overline{C}_{n_r} r\right)\right] +$$

$$\overline{q}SbI_z\left[\overline{C}_{l_\beta}\beta + \frac{b}{2(U_0 + u)}\left(\overline{C}_{l_p} p + \overline{C}_{l_r} r\right)\right]\Bigg\} (A.32)$$

$$M_1 = \frac{1}{I_y}\Bigg\{\left[-rp\left(I_x - I_z\right) - I_{xz}\left(p^2 - r^2\right)\right] +$$

$$\overline{q}S\overline{c}\begin{bmatrix} \overline{C}_{m_u}\dfrac{u}{(U_0 + u)} + \overline{C}_{m_\alpha}\alpha + \\ \dfrac{\overline{c}}{2(U_0 + u)}\overline{C}_{m_q} q + \overline{C}_{m_{\delta_c}}\delta_c \end{bmatrix}\Bigg\} (A.33)$$

$$N_1 = \frac{1}{I_x I_z - I_{xz}^{\ 2}} \Bigg\{ \begin{bmatrix} -qrI_{xz}\left(I_x - I_y + I_z\right) - \\ pq\left(I_x I_y - I_x^{\ 2} - I_{xz}^{\ 2}\right) \end{bmatrix} +$$

$$\overline{q}SbI_{xz}\left[\overline{C}_{l_\beta}\beta + \frac{b}{2(U_0 + u)}\left(\overline{C}_{l_p} p + \overline{C}_{l_r} r\right)\right] +$$

$$\overline{q}SbI_x\left[\overline{C}_{n_\beta}\beta + \frac{b}{2(U_0 + u)}\left(\overline{C}_{n_p} p + \overline{C}_{n_r} r\right)\right]\Bigg\} (A.34)$$

Equations of Motion

$$\dot{p} = \frac{1}{I_x I_z - I_{xz}^{\ 2}} \Bigg\{ \begin{bmatrix} pqI_{xz}\left(I_x - I_y + I_z\right) - \\ qr\left(I_z^{\ 2} - I_y I_z + I_{xz}^{\ 2}\right) \end{bmatrix} +$$

$$\overline{q}SbI_{xz}\begin{bmatrix} C_{n_\beta}\beta + \dfrac{b}{2(U_0 + u)}\left(C_{n_p} p + C_{n_r} r\right) + \\ C_{n_{\delta_a}}\delta_a + C_{n_{\delta_t}}\delta_t + C_{n_{\delta_{dc}}}\delta_{dc} + C_{n_{\delta_r}}\delta_r \end{bmatrix} +$$

$$\overline{q}SbI_z\begin{bmatrix} C_{l_\beta}\beta + \dfrac{b}{2(U_0 + u)}\left(C_{l_p} p + C_{l_r} r\right) + \\ C_{l_{\delta_a}}\delta_a + C_{l_{\delta_t}}\delta_t + C_{l_{\delta_{dc}}}\delta_{dc} + C_{l_{\delta_r}}\delta_r \end{bmatrix}\Bigg\} (A.35)$$

$$\dot{q} = \frac{1}{I_y}\left\{\left[-rp\left(I_x - I_z\right) - I_{xz}\left(p^2 - r^2\right)\right] + \bar{q}S\bar{c}\begin{bmatrix} C_{m_u}\dfrac{u}{\left(U_0+u\right)} + C_{m_\alpha}\alpha + \\ \dfrac{\bar{c}}{2\left(U_0+u\right)}C_{m_q}q + C_{m_{\delta_s}}\delta_s + C_{m_{\delta_c}}\delta_c \end{bmatrix}\right\} \quad (A.36)$$

$$\dot{r} = \frac{1}{I_xI_z - I_{xz}{}^2}\left\{\begin{bmatrix} -qrI_{xz}\left(I_x - I_y + I_z\right) - \\ pq\left(I_xI_y - I_x{}^2 - I_{xz}{}^2\right) \end{bmatrix} + \bar{q}SbI_{xz}\begin{bmatrix} C_{l_\beta}\beta + \dfrac{b}{2\left(U_0+u\right)}\left(C_{l_p}p + C_{l_r}r\right) + \\ C_{l_{\delta_a}}\delta_a + C_{l_{\delta_t}}\delta_t + C_{l_{\delta_{dc}}}\delta_{dc} + C_{l_{\delta_r}}\delta_r \end{bmatrix} + \bar{q}SbI_x\begin{bmatrix} C_{n_\beta}\beta + \dfrac{b}{2\left(U_0+u\right)}\left(C_{n_p}p + C_{n_r}r\right) + \\ C_{n_{\delta_a}}\delta_a + C_{n_{\delta_t}}\delta_t + C_{n_{\delta_{dc}}}\delta_{dc} + C_{n_{\delta_r}}\delta_r \end{bmatrix}\right\} \quad (A.37)$$

$$\dot{u} = -g\sin\Theta\cos\Theta_0 + rv + \frac{\bar{q}S}{m}\begin{bmatrix} C_{x_u}\dfrac{u}{\left(U_0+u\right)} + C_{x_\alpha}\alpha + \\ C_{x_{\delta_s}}\delta_s + C_{x_{\delta_c}}\delta_c \end{bmatrix} \quad (A.38)$$

$$\dot{v} = g\sin\Phi\cos\Theta_0 - \left(U_0+u\right)r + pw + \frac{\bar{q}S}{m}\begin{bmatrix} C_{y_\beta}\beta + \\ C_{y_{\delta_a}}\delta_a + C_{y_{\delta_t}}\delta_t + C_{y_{\delta_{dc}}}\delta_{dc} + C_{y_{\delta_r}}\delta_r \end{bmatrix} \quad (A.39)$$

$$\dot{w} = -g\sin\Theta\sin\Theta_0 + \left(U_0+u\right)q - pv + \frac{\bar{q}S}{m}\begin{bmatrix} C_{z_u}\dfrac{u}{\left(U_0+u\right)} + C_{z_\alpha}\alpha + \\ C_{z_{\delta_s}}\delta_s + C_{z_{\delta_c}}\delta_c \end{bmatrix} \quad (A.40)$$

References

[1] Kim, B. S. and Calise, A. J., "Nonlinear Flight Control Using Neural Networks", AIAA Paper 94-3646-CP, 1994.

[2] Chiang, C. and Youssef, H. M., "Neural Network Approach to Aerodynamic Coefficients Estimation and Aircraft Failure Isolation Design, AIAA Paper 94-3599-CP, 1994.

[3] Napolitano, M. R. , Naylor, S., Neppach, C., and Casdorph, V., "On-Line Learning Non-Linear Direct Neuro Controllers for Restructurable Control Systems", AIAA Paper 94-3643-CP, 1994.

[4] Barron, R. L., Cellucci, R. L., Jordan, P. R. III, and Beam, N. E., "Applications of Polynomial Neural Networks to FDIE and Reconfigurable Flight Control", IEEE Paper CH2881-1/90/0000-0507, 1990.

[5] Zeh, J. M., Young, D. L., and Couture, N. J., "STOL and Maneuver Technology Demonstrator (S/MTD) Stability Derivatives", WL-TR-92-3055, 1992.

[6] Kim, B. S., "Nonlinear Flight Control Using Neural Networks", PhD. Thesis, Georgia Institute of Technology, School of Aerospace Engineering, 1993.

[7] Totah, J., "An Examination of Aircraft Aerodynamic Estimation Using Neural Networks", SAE Technical Paper Series 952036, 1995.

Figure 1. F-15 ACTIVE Aircraft

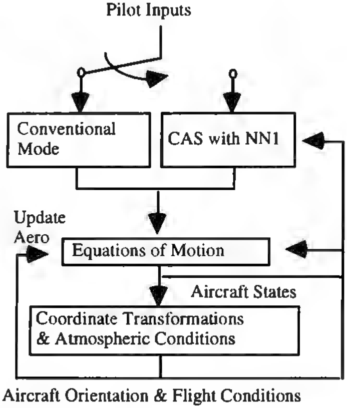
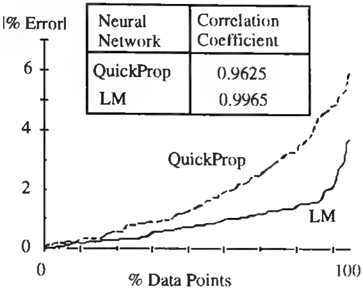

Pilot Inputs
Conventional Mode
CAS with NN1
Update Aero
Equations of Motion
Aircraft States
Coordinate Transformations & Atmospheric Conditions
Aircraft Orientation & Flight Conditions

Figure 2. F-15 ACTIVE Simulation

Figure 3. Simulator POD and F-15 ACTIVE Graphics.

$$\begin{tikzcd}
\begin{bmatrix}\delta_{la}\\ \delta_{ln}\\ \delta_{pd}\end{bmatrix} \arrow[r, "f1"] & \begin{bmatrix}\dot{\Phi}_c\\ \dot{\Theta}_c\\ \dot{\Psi}_c\end{bmatrix} \arrow[r, "f2"] & \begin{bmatrix}U_1\\ U_2\\ U_3\end{bmatrix} \arrow[r, "f3"] & \begin{bmatrix}\dot{P}_c\\ \dot{Q}_c\\ \dot{R}_c\end{bmatrix} \arrow[r, dashed, "f4"] & \begin{bmatrix}\delta_t\\ \delta_s\\ \delta_r\end{bmatrix}
\end{tikzcd}$$

Figure 4. Direct Adaptive Tracking Controller.



Neural Network	Correlation Coefficient
QuickProp	0.9625
LM	0.9965

Figure 5. NN1 Error Histograms for C_{Z_α} .

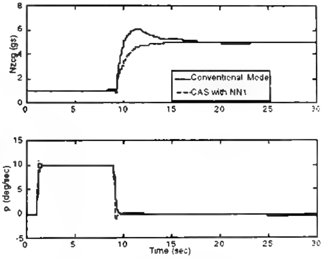


Figure 6. Normal Acceleration and Roll Rate Responses to Stick Inputs; Mo=0.6, ho=10,000 ft, 100% Throttle, Nominal Conditions.

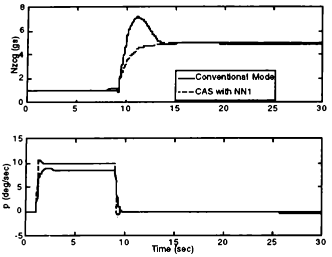


Figure 7. Normal Acceleration and Roll Rate Responses to Stick Inputs; Mo=0.6, ho=10,000 ft, 100% Throttle, 30% Loss of Effectiveness to all Control Surfaces, Condition Unknown to NN1.

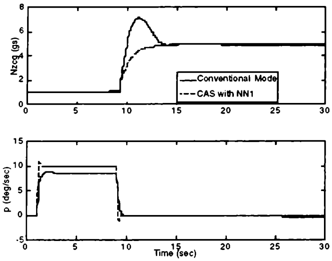


Figure 8. Normal Acceleration and Roll Rate Responses to Stick Inputs; Mo=0.6, ho=10,000 ft, 100% Throttle, 30% Loss of Effectiveness to all Control Surfaces, Condition Known to NN1.

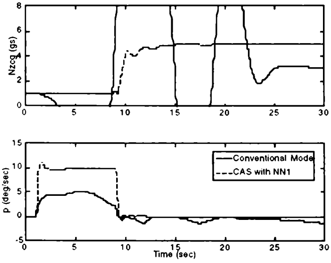


Figure 9. Normal Acceleration and Roll Rate Responses to Stick Inputs; Mo=0.6, ho=10,000 ft, 100% Throttle, 80% Loss of Effectiveness to all Control Surfaces, Condition Unknown to NN1.

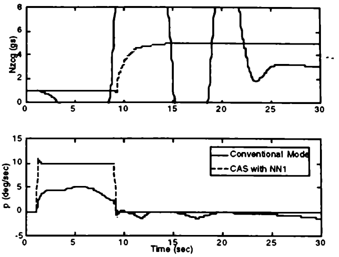


Figure 10. Normal Acceleration and Roll Rate Responses to Stick Inputs; Mo=0.6, ho=10,000 ft, 100% Throttle, 80% Loss of Effectiveness to all Control Surfaces, Condition Known to NN1.

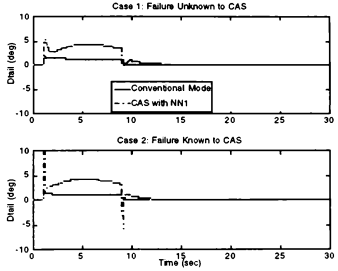


Figure 11. Differential Stabilator Responses to Stick Inputs; Mo=0.6, ho=10,000 ft, 100% Throttle, 80% Loss of Effectiveness to all Control Surfaces.

AIAA-96-3504-CP

A HIGHER FIDELITY POINT-MASS SIMULATION OF AIRCRAFT DYNAMICS

Conrad Mukai, Engineer & Software Consultant
San Jose, CA
conrad@best.com

George Hunter, Project Manager
Seagull Technology, Inc.
21771 Stevens Creek Boulevard
Cupertino, CA 95014-1175

Abstract

This paper presents the development of an aircraft dynamics modeling and trajectory simulation methodology which is currently in use at NASA and FAA laboratories. The initial objectives of the work were to provide the simulation kernel for a real-time radar tracking simulation. The simulated radar returns are displayed on an air traffic controller's plan view display. A key requirement was to achieve the appropriate balance of fidelity and execution speed. Sufficient fidelity was required so test subjects would not be distracted by unrealistic aircraft tracks. Sufficient speed was required for the real-time simulation of hundreds of aircraft simultaneously. Since the initial development, additional modeling capabilities have been added and the tool, referred to as *Amelia*, has been used in a variety of applications. These include: flight efficiency studies, flight technical error evaluations, Monte Carlo simulations for flight time variation investigations, air traffic control automation analyses, conflict detection and resolution in multi-aircraft encounters, free-flight studies, and flight planning.

1 Introduction

When choosing between aircraft trajectory simulation algorithms, tradeoffs between fidelity, speed, and cost are important issues. Sophisticated cockpit simulators as well as 6 Degrees Of Freedom (DOF) computer simulations accurately predict aircraft dynamics, but they are costly to develop, acquire, use, maintain and run. These simulators are expensive but are required by high-detail applications such cockpit human-factors or control system inner-loop design projects. At the other extreme, 3 DOF point-mass simulations are an alternative when cost, multiple aircraft modeling, and speed are more critical than precise representation. The problem with these simplified methods is that important real-world aspects can be ignored. In many applications, the best solution is a compromise in which improved modeling is obtained for a relatively small sacrifice in cost and speed. In this paper a modeling technology is presented which incorporates the speed and affordability of a point-mass system with the sophisticated modeling of real-world transients.

1.1 Motivation

The Center TRACON Automation System (CTAS) project is an on going effort led by the NASA Ames Research Center to develop and implement automated air traffic control tools. CTAS uses aircraft radar tracking data and wind measurements to provide traffic managers with a schedule of arrival times and provide controllers with suggested advisories to meet those schedules. As part of the CTAS validation, NASA Ames had air traffic controllers evaluate the various tools. During these real-time simulations, the controllers were distracted by the unrealistic movements of the radar returns on the simulated radar Plan View Displays (PVDs). These radar tracks were generated by a 3 DOF point-mass simulation algorithm. Aircraft turns and descents to be executed too precisely and too quickly. From these tests arose the need for a higher fidelity real-time trajectory simulation capable of handling tens or even hundreds of aircraft. Seagull Technology, Inc. developed the aircraft dynamics modeling system *Amelia* to meet these needs.

1.2 Algorithm

Piloting tendencies and variations are an important aspect in the real world. If realistic trajectories are required (as opposed to optimum or maximum maneuvers, for example), then the higher fidelity provided by a 6 DOF simulation is of little use if a pilot model is not included. The term "pilot model" is used here not in the short-period or inner-loop sense, but rather as a simulator of how pilots capture and maintain routes, altitudes, and speeds — the outer loop. *Amelia* incorporates this sort of pilot model. In addition, error modeling has been incorporated into *Amelia* to produce even more realistic distributions of trajectories. A radar track generated by *Amelia* will show variations representative of actual radar tracks.

Amelia is a kernel simulation well suited for multiple aircraft modeling (150 have been simulated simultaneously in real-time on a HP 9000-710 workstation). *Amelia* implements a control loop around the non-linear point-mass equations of motion to produce realistic equations of motion . In addition to 3-D translation, the aircraft roll orientation is modeled to provide a 4 DOF capability.

In *Amelia*, several methods are available to control horizontal- and vertical-plane motion. Additionally,

Presented at the 1996 AIAA Guidance, Navigation and Control Conference, San Diego, CA.

Amelia can be commanded to follow a TRACON route defined by giving one waypoint the property of being an Instrument Landing System (ILS) beam. The ILS beam is a waypoint that describes the aircraft's final approach for a landing. The vertical control of aircraft is controlled by selection of a target altitude and speed. The control law continuously adjusts the thrust and coefficient of lift to converge to the desired speed / altitude state.

Amelia was originally developed in the C programming language, and has been updated to a C++ library version. C++ provides *Amelia* with an object oriented architecture which greatly simplifies many of the programming aspects such as creating new aircraft, reusing routes, and selecting weather conditions.

2 Equations of Motion for a Point Mass Aircraft

In order to effectively simulate the aircraft behavior, *Amelia* integrates the equations of motion for a point mass aircraft. The point mass equations represent the translational dynamics of the aircraft, since as the name implies, the only inertial property considered is mass. In addition to the three degrees of freedom (DOF) associated with aircraft position, a fourth DOF for the aircraft roll orientation relative to the horizon has been included to add further realism to the simulation.

2.1 Forces

The underlying assumptions made in using the point mass dynamics reveal a physical interpretation to the equations. Essentially, three force vectors fixed in the body frame of the aircraft and a fourth vector fixed in the inertial frame drive the aircraft motion. The body-fixed forces are thrust T, which is directed along the positive $\mathbf{x}_b$ axis, lift L directed along the negative $\mathbf{z}_b$ axis, and drag D parallel to the negative $\mathbf{x}_b$ axis. The pilot alters the orientations and magnitudes of these three forces to control the translational motion of the aircraft. The fourth force, weight W is always directed downward, along the negative $\mathbf{z}_e$ axis. In addition to these forces, apparent forces in the form of accelerations due to wind shear act as external disturbances to the aircraft. These terms are referred to as wind gradient factors.

In a real aircraft, pitching the nose of the aircraft alters the lift. *Amelia* simulates this effect by adjusting the coefficient of lift C_L (which in turn alters the aircraft flight path angle γ_a) to drive the aircraft altitude to the commanded value. In addition, the coefficient of lift and the aircraft configuration parameters, namely the flap angle δ_f, the landing gear position δ_g, and the spoiler setting δ_s, determine the coefficient of drag C_D. The coefficient of drag computation is based on aircraft manufacturer's performance data. Using aircraft performance data insures that the ratio of lift to drag is realistic, resulting in accurate descent profiles.

The other control force, thrust, drives the aircraft speed to the desired value. To insure that the thrust is reasonable, the thrust level computed by the control law is checked against minimum and maximum values from manufacturer's engine data. Note that both the control parameters, coefficient of lift and thrust, are set instantaneously, within bounds, to specified values. This means that the short period and engine spooling dynamic lags which occur in real aircraft are ignored in the simulation. The relative slowness of the translational dynamics compared to the dynamics associated with pitching the aircraft or with engine spooling justifies this simplification.

The aerodynamic forces, lift and drag, relate to the control parameters, C_L, δ_f, δ_g and δ_s through the following equations:

$$\begin{aligned} L &= C_L\, q\, S, \qquad D = C_D\, q\, S, \\ C_D &= C_D\left(M, C_L, \delta_f, \delta_g, \delta_s\right), \\ q &= \frac{\rho V_a^2}{2}, \qquad V_a = a\, M. \end{aligned} \tag{1}$$

2.2 Dynamic Equations of Motion

The dynamic equations of motion relate the external forces to the aircraft velocity in the inertial frame. The aircraft velocity with respect to the inertial frame is composed of two parts, the velocity of the aircraft relative to the air mass and the wind velocity at the aircraft current position. The state variables (true airspeed V_a, the flight path angle γ_a, and the heading ψ_a) describe the aircraft velocity relative to the air mass. Both angle of attack and side slip are ignored. Additional terms associated with the wind gradient (i.e., the change in wind velocity with changes in altitude) appear on the right-hand side of the equations. The dynamic equations of motion are listed below:

$$\begin{aligned} \dot{V}_a &= \frac{T - D - W \sin\gamma_a}{m} - W_{igf} V_a \cos\gamma_a \sin\gamma_a \\ \dot{\gamma}_a &= \frac{L\cos\phi - W \cos\gamma_a}{mV_a} + W_{igf} \sin^2\gamma_a \\ \dot{\psi}_a &= \frac{L\sin\phi}{mV_a\cos\gamma_a} - W_{cgf} \tan\gamma_a \end{aligned} \tag{2}$$

The wind gradient factors, W_{cgf} and W_{igf} account for the cross-track and in-track acceleration effects.

2.3 Kinematic Equations of Motion

The kinematic equations of motion relate the velocity vector to the aircraft position coordinates. The coordinates x and y specify the aircraft ground position in the inertial frame. The state variable h is the aircraft altitude. *Amelia* integrates the following differential equations to propagate the aircraft position.

$$\begin{aligned} \dot{h} &= V_a \sin \gamma_a \\ \dot{x} &= V_a \cos \gamma_a \sin \psi_a + V_{wx} \\ \dot{y} &= V_a \cos \gamma_a \cos \psi_a + V_{wy} \end{aligned} \tag{3}$$

2.4 Roll Dynamics

To turn an aircraft, a pilot must roll the aircraft to point the lift vector in the direction of the turn. This effect can be seen in Eqs. (2). If the wind gradient term is ignored, the turn rate becomes directly proportional to sin ϕ. Hence, as roll angle increases so does turn rate. Because of this property, ϕ could be used as a control variable; however, unlike the pitch dynamics or engine spooling, the roll motion of an aircraft to produce a turn is gradual enough to have a noticeable effect on translational motion. To further complicate the issue, aerodynamic modeling of an aircraft rotational dynamics would increase the computational load tremendously.

To avoid lengthy calculations, a simpler model of the roll motion is used, since the lag associated with roll need only be approximated to generate realistic trajectories. In order to accommodate the need for a simple dynamic model, *Amelia* models the roll motion with the following second order system of linear differential equations.

$$\begin{aligned} \dot{\phi} &= \omega \\ \dot{\omega} &= \frac{1}{\tau_\phi} \left[K_\phi \left(\phi_r - \phi \right) - \omega \right] \end{aligned} \tag{4}$$

The new variable ϕ_r is the control parameter. The roll gain K_ϕ and the roll time constant τ_ϕ are set to approximate the typical roll response of a commercial jet aircraft. The values currently being used are 0.5102 sec^{-1} and 1.00 sec, respectively. These values correspond to a natural period of 8.8 sec and a damping ratio of 0.7.

The relatively fast roll equations exhibit poor behavior when advanced with integration time steps much greater than 0.5 sec. To avoid requiring a smaller time step, an explicit solution of the linear roll equations is used. A zero-order hold on the discrete values of ϕ_r is assumed to generate the solution. The roll and roll rate equations of motion then converts to the constant coefficient difference equation over the integration period Δt.

3 Control Law Design

The control law in *Amelia* is based upon a linear model of the point mass aircraft equations. Linearizing the point mass equations reveals two types of uncoupled motion, the lateral motion and the vertical motion. The control law follows the same break down. The lateral controller determines the response to heading commands, while the vertical, or speed-altitude, controller defines the dynamics associated with the aircraft phugoid mode. This section focuses primarily upon the method of gain selection for both of these controllers. The non-linearities and mode transitioning of the control law are discussed in Sec. 4.

3.1 Linear Model of Lateral Equations

The state variables ψ_a, ϕ, and ω and the control parameter ϕ_r define the lateral dynamics. To form a linear model of this system, care must be taken to account for the difference between the integration scheme used to advance the roll equations and the Euler's integration used to advance the remaining state variables. Because roll and roll rate are advanced with an explicit solution, they can be thought of as a continuous system which is periodically being sampled. The Euler's integration on the other hand converts Eqs. (2) and (3) to a set of recurrence equations. Therefore, these equations should be viewed as being purely discrete. This distinction is important because it accounts for the dependence of lateral stability upon step size.

To linearize the lateral system, it is assumed that the aircraft is in level flight and at constant speed. Applying these conditions to Eqs. (2) results in the following:

$$\dot{\psi}_a = \frac{g \tan \phi}{V_a} \tag{5}$$

Linearizing this equation and discretizing it with an Euler's integration scheme results in the following difference equation:

$$\psi_a(k+1) = \psi_a(k) + \frac{g\,\Delta t}{V_a}\,\phi(k) \qquad (6)$$

Equation (6) is then placed in series with the continuous version of the roll equations—see Eqs. (4). Appropriate sample and hold elements are added to produce a linear model of the heading dynamics.

A proportional feedback loop is used to control the heading. Figure 3.1 shows a block diagram of this. From the block diagram it can be seen that heading error is fed back with a proportional gain to determine the commanded roll angle, ϕ_r. This in turn is passed through the roll dynamics to produce the roll angle which drives the aircraft heading.

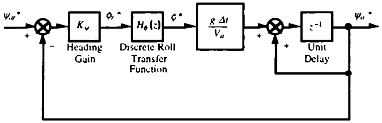


Figure 3.1 Block diagram of the linear heading model. Asterisk denotes z-transform.

The block labeled "Discrete Roll Transfer Function" in Fig. 3.1 has the continuous version of the roll dynamics embedded in it. Figure 3.2 is an expanded view of this block. It shows how the continuous equations are made compatible with the discrete system surrounding it. In Fig. 3.2, the discrete values of ϕ_r are held constant over the integration period Δt and input into the continuous roll dynamics equations. Then the output is sampled at the end of the integration period and passed to the discrete system while a new value for ϕ_r is being read in to repeat the process.

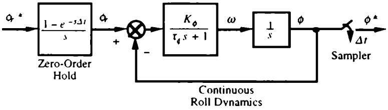


Figure 3.2 Conversion of continuous roll dynamics to a discrete system.

It is interesting that this is the reverse of typical digital control systems applications. Normally, the output of the analog plant is sampled and passed through a digital controller. The controller output is then held in a D/A converter which in turn loops back into the analog system. Here, the output of the digital plant, namely Eq. (6), is held and then processed through an analog set of lags. This analog response is then sampled to generate a discrete control output.

The block diagram in Fig. 3.1 reveals that the heading dynamics depend upon the true airspeed of the aircraft. High-speed heading capture is sluggish, while low-speed capture is less stable. In order to compensate for this, *Amelia* uses a variable gain to normalize the dynamic behavior. By making the feedback gain proportional to the true airspeed, the dynamic response at all speeds becomes consistent. Equation (7) defines the normalized gain L_ψ which is used in *Amelia* to regulate the feedback.

$$K_\psi = L_\psi V_a \qquad (7)$$

3.2 Gain Selection for the Lateral Controller

In addition to true airspeed, the other parameter which has a significant effect upon the dynamics is the time step Δt. If the time step is small compared to the time constant of the roll equations (not to be confused with τ_ϕ), which is on the order of 6 sec, then the delay in the commanded roll angle will not have a significant effect on the heading dynamics; however, as the time step is set to values approaching the roll time constant, the delay in the response to the commanded roll angle tends to destabilize the system.

Because the relationship between Δt and the dynamics of the discrete roll transfer function is complicated, a closed-form equation relating the time step to the feedback gain is not possible. In lieu of a mathematical expression relating time step to normalized feedback gain, Table 3.1 contains a list of recommended gains for several time steps and the associated settling times.

Table 3.1 Recommended gains for various time steps and their settling times.

Δt (sec)	L_ψ (10^{-3} sec-ft^{-1})	Settling time
0.5	5.4	8.5
1.0	4.4	11.1
2.5	3.1	14.6
5.0	1.9	23.9

3.3 Reference Solution for the Vertical Control Law

The vertical control law in *Amelia* drives the aircraft speed and altitude to their respective commanded values. The control law accomplishes this by continually adjusting C_L and T so that speed and altitude converge upon the desired values. In addition to meeting the commanded speed and altitude, the controller in *Amelia* must also maintain a constant speed when going from one altitude to another to realistically reproduce the trajectories of commercial aircraft. For a given throttle setting, the control parameter values which produce a constant speed trajectory are a function of altitude. In this report these control parameter values are referred to as the reference solution.

During a descent, T is usually set to idle or to some value which produces a desired descent rate. In either case this leaves C_L as the primary control variable during descents. To determine the C_L value, one of the following equations is used:

$$C_L = C_{Lr} + K_{IAS}(V_{ir} - V_i) + K_{\gamma, IAS}(\gamma_{ar} - \gamma_a), \tag{8}$$

$$C_L = C_{Lr} + K_M(M_r - M) + K_{\gamma, M}(\gamma_{ar} - \gamma_a), \tag{9}$$

$$C_L = C_{Lr} + K_\gamma(\gamma_{ar} - \gamma_a). \tag{10}$$

The subscripted K's in Eqs. (8) through (10) are the feedback gains. The computation of gains is covered in Secs. 3.5 through 3.7. In this section, the discussion focuses on the variables subscripted with an r, which describe the aforementioned reference solution.

Each of the above equations applies to a specific phase of control applied to the aircraft. Equations (8) or (9) are used when a commanded speed, specified as either indicated airspeed or Mach number, respectively, is to be captured and maintained. Equation (10) is used during transitions between speeds, such as a deceleration at level flight. In either case, a feedback loop drives the aircraft speed and/or the flight path angle to the desired trajectory. These target values are called the reference indicated airspeed V_{ir}, the reference Mach number M_r and the reference flight path angle γ_{ar}. In addition to these feedback terms, the controller requires a bias value called the reference coefficient of lift C_{Lr} since C_L does not go to zero for null feedback errors.

In general, to compute the reference parameters for a specified maneuver, a steady state condition is invoked by setting the dynamic equations of motion or some form of them, to zero. Then, by using the commanded value of speed, the current altitude and the bank angle the reference parameters can be determined.

3.4 Linearization of Models for Selection of Vertical Controller Gains

Computation of the feedback gains shown in Eqs. (8) through (10) requires a linear model of the vertical plane dynamics. By assuming that the bank angle is equal to zero and linearizing Eqs. [2] and [3] about a particular reference solution, a state space model is generated. The state variables of this model are the deviations in true airspeed, flight path angle and altitude from the reference solution. The input to the model is the deviation of the coefficient of lift from its reference value. In the actual gain computation, the linearization process is performed numerically; however, expressions for all the state space terms can be derived by expanding the differential equations in Taylor series form and retaining all terms up to first-order in the state and control variables. This is not done here since a derivation or a listing of linearized equations does not enhance the understanding of this problem.

To use the state space model for gain selection requires careful examination of the feedback law used to control the aircraft. Both C_{Lr} and γ_{ar} depend upon altitude for the constant indicated airspeed and the constant Mach number descents. In addition, in the constant vertical speed descent, C_{Lr} and γ_{ar} also depend upon true airspeed. This means that even when all feedback gains are zero, there remains a dependency of the control variable C_L upon the state of the aircraft through the reference solution computation of C_{Lr}. This implies that the "open-loop" model used in the linear analysis must account for this effect.

Define the state space vector $\mathbf{x}$ as follows:

$$\mathbf{x} = \begin{bmatrix} V_a \\ \gamma_a \\ h \end{bmatrix}. \tag{11}$$

Then the linearization of Eqs. (2) and (3) results in the standard format for linear state equations:

$$\dot{\mathbf{x}} = \mathbf{A}_0\mathbf{x} + \mathbf{b}C_L. \tag{12}$$

In Eq. (12), $\mathbf{A}_0$ contains the partial derivatives of the non-linear equations of motion with respect to the state variables and $\mathbf{b}$ contains the partial derivatives of the same equations of motion with respect to C_L. Now to account for the dependency of C_{Lr} upon the state variables, the following vector of partial derivatives can be computed numerically:[1]

$$\frac{\partial C_{Lr}}{\partial \mathbf{x}} = \begin{bmatrix} \dfrac{\partial C_{Lr}}{\partial V_a} & \dfrac{\partial C_{Lr}}{\partial \gamma_a} & \dfrac{\partial C_{Lr}}{\partial h} \end{bmatrix}. \tag{13}$$

Note that for all the reference solutions discussed in Sec. 3.3 the partial derivative $\partial C_{Lr} / \partial \gamma_a$ is equal to zero, but for the sake of generality it is included in Eq. (13). Feeding back the influence of C_{Lr} on the control variable C_L, results in a new state matrix:

$$\mathbf{A} = \mathbf{A}_0 + \mathbf{b}\frac{\partial C_{Lr}}{\partial \mathbf{x}}. \tag{14}$$

The system defined by **A** and **b** now takes into account the first-order effects of the reference solution in the absence of any feedback gains.

3.5 Model Reduction for Speed Control

Table 3.4 contains the poles and zeros of an open-loop state space model linearized about the reference solution of a B727-200 on a 280 kt idle descent at 35000 ft.

Table 3.4 Typical poles and zeros for an idle descent model.

Poles (sec^{-1})	Transmission zeros for V_i (sec^{-1})	Transmission zeros for γ_a (sec^{-1})
$(-0.56 \pm 6.52i) \times 10^{-2}$ -1.53×10^{-3}	-8.84×10^{-1} -1.53×10^{-3}	-5.24×10^{-3} -1.23×10^{-3}

The complex conjugate poles in Table 3.4 represent the phugoid dynamics. The low magnitude of the real part, relative to the imaginary part, of the phugoid poles reflects the low damping of the open-loop phugoid mode (this example has a damping ratio of 0.086). The other pole in the table originates from the dynamics associated with changes in altitude. Its relatively long time constant reflects the slow nature of this phenomena. The proximity of the altitude pole to transmission zeros in both V_i and γ_a suggests that the model can be reduced to second order. A reduced order model not only simplifies the equations but also allows for the use of full state feedback in the pole placement procedures since only two gains are used in the speed control laws.

If a pole-zero cancellation approach is used, then the reduced transfer functions from C_L to V_i and from C_L to γ_a can be written as:

$$\begin{aligned} H_{IAS}(s) &= \frac{k_{IAS}(s + \sigma_{IAS})}{s^2 + 2\zeta_p\omega_p s + \omega_p^2} \quad \text{and} \\ H_{\gamma}(s) &= \frac{k_{\gamma}(s + \sigma_{\gamma})}{s^2 + 2\zeta_p\omega_p s + \omega_p^2}. \end{aligned} \tag{15}$$

In Eqs. (15), k_{IAS} and k_{γ} are the static gains, σ_{IAS} and σ_{γ} are the uncanceled zeros and ζ_p and ω_p are the open-loop phugoid damping ratio and natural frequency. The two transfer functions in Eqs. (15) can be combined into a SIMO (single input, multi-output) state space model:

$$\begin{aligned} \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} &= \begin{bmatrix} -2\zeta_p\omega_p & -\omega_p^2 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} C_L, \\ \begin{bmatrix} V_i \\ \gamma_a \end{bmatrix} &= \begin{bmatrix} k_{IAS} & k_{IAS}\sigma_{IAS} \\ k_{\gamma} & k_{\gamma}\sigma_{\gamma} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}. \end{aligned} \tag{16}$$

The state variables, x_1 and x_2, in this realization do not have an obvious correspondence to any physical parameters. To make this model compatible with variables used in the feedback law, the model can be transformed by using a new state vector. Referring to the matrices in Eq. (16) as $\mathbf{A}_{r0}$, $\mathbf{b}_{r0}$ and $\mathbf{c}_{r0}$, then a transformation to the state vector $[V_i \ \gamma_a]^{\mathrm{T}}$ can be performed with the following operations:

$$\begin{aligned} \mathbf{A}_r &= \mathbf{c}_{r0}\mathbf{A}_{r0}\mathbf{c}_{r0}^{-1} \quad \text{and} \\ \mathbf{b}_r &= \mathbf{c}_{r0}\mathbf{b}_{r0}. \end{aligned} \tag{17}$$

Pole-placement using full state feedback can be applied to $\mathbf{A}_r$ and $\mathbf{b}_r$ to determine the gains for a desired set of closed-loop dynamics. The linearization and reduction process described in the previous section and in this section can be repeated for the Mach number feedback case.

3.6 Pole Placement for Speed Control Gains

There are various methods for the placement of closed-loop poles. The technique used for this particular application is the Bass-Gura formula described in Ref. [1]. The Bass-Gura formula requires that all the state variables in the system are controllable. For this particular application, a system is considered controllable if the controllability matrix is full rank.

In essence, the pole placement process can be thought of as setting the coefficients from the following determinant equation,

$$\left| s\mathbf{I} - \mathbf{A}_r + \mathbf{b}_r \begin{bmatrix} K_{IAS} & K_{\gamma} \end{bmatrix} \right| = 0, \tag{18}$$

equal to the desired closed-loop characteristic equation,

$$s^2 + 2\zeta_c\omega_c s + \omega_c^2 = 0. \tag{19}$$

Here, ζ_c and ω_c are the desired closed-loop damping ratio and natural frequency. This results in two equations from the coefficients of s in Eq. (19) and two unknowns, K_{IAS} and K_γ.

For linear systems, pole placement allows the closed-loop poles to be located anywhere in the s-plane; however, this does not hold true for non-linear systems due to the limits imposed by the dynamic range of the control variable. In *Amelia*, the dynamic range of C_L has a maximum bound imposed by the stall value. In addition, although there is no minimum bound in *Amelia*, large negative values of C_L should be avoided to maintain realism. These limitations prevent the gains from becoming excessively large.

The root locus method of Evans, described in Ref. [2], helps to visualize the correlation between the feedback gains and the closed-loop dynamics. In Fig. 3.8, three root-loci are shown. The outer dashed circle represents the locus of closed-loop poles resulting from feedback of indicated airspeed only. The small dashed inner arc represents poles resulting from feedback of flight path angle only. Note that the poles and zeros associated with the dashed root loci correspond to the poles and zeros listed in Table 3.4. The third root locus, denoted by the solid circle, corresponds to a feedback law using both indicated airspeed and flight path angle. By examining the characteristics of each of these loci, we can develop a good feel for the feedback laws described in Eqs. (8) and (9).

The outer root locus corresponding to feedback of V_i alone shows that this control law is very responsive (the farther a pole is from the origin the faster its rèsponse). For example, if a point along the circle corresponding to a damping ratio of 0.7 is chosen as the design (i.e., a complex pair at $-0.86 \pm 0.88i$), then the natural frequency of the system is 1.23 sec $^{-1}$. The time constant for a second order system is defined as the inverse of the product of the damping ratio and the natural frequency, so the time constant of this system is 1.16 sec. For a step command, a second-order system moves within 2% of the commanded value after 4 time constants, which translates to a relatively quick capture time of about 4.6 sec in this example. As previously mentioned, a fast response in the linear model may require an unrealistic dynamic range for the control variable. That is the case here. For this particular example, the feedback gain which produces the desired dynamics is –0.51 sec-ft $^{-1}$. Multiplying this gain by a relatively small 10 kt speed error results in a coefficient of lift feedback of 8.6, which places C_L well outside the range of realistic values. Reducing the feedback gain magnitude to decrease the required dynamic range of C_L produces less than acceptable performance. Setting the gain to a value of –0.1 sec-ft $^{-1}$ results in a damping ratio of 0.32.

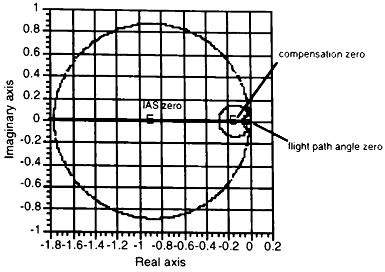


Figure 3.8 Root loci for speed control.

The inner root locus corresponding to feedback of γ_a alone is sluggish. The zero located near the origin attracts one branch of the root locus, which results in a slower response for higher gains. For example, with a gain of 0.8, the closed-loop dynamics for this control law has a second-order time constant of 20.9 sec. For a gain of 10.0, the two roots lie upon the real axis, resulting in two first-order poles. The slower of the two poles has a time constant of 109.6 sec. Time constants of this order are poor models of the speed capture dynamics.

Feeding back both indicated airspeed and flight path angle compromises the extremes of using single variable feedback. In Fig. 3.8 the solid circle is the root locus of a control law where both V_i and γ_a are used. Note that the size of the circle is dictated by the location of the compensation zero. The location of the zero in turn is determined by the ratio of the indicated airspeed gain to the flight path angle gain. For a ratio near negative infinity, the zero moves out to the position denoted by the label "IAS zero." For low ratios, the compensation zero moves towards the origin. Hence, the size and shape of the root locus shifts from the outer circle for pure indicated airspeed feedback to the inner arc for pure flight path angle feedback. The design point along the circle is determined by the actual value of the gains (as opposed to the ratio of the gains which sets the shape of the root locus). For example, a design criteria of 0.7 damping ratio and 0.2 sec $^{-1}$ natural frequency requires feedback gains of –0.0117 sec-ft $^{-1}$

for indicated airspeed and 2.17 for flight path angle. The C_L term for a 10 kt indicated airspeed error is 0.20, the term for a 3° flight path angle error is 0.11, so dynamic range is not a problem. Also, the design dynamics specified produce reasonable capture times. For the damping ratio and natural frequency chosen, the time constant is 7.1 sec, which is a reasonable value.

The process described above determines the gains for a specific speed and altitude. The control law insures that speed remains close to the value about which the design model is linearized. Altitude, on the other hand, could be constantly changing since in many cases speed commands are accompanied by a descent to a new altitude. To avoid recomputing the gains for changing altitudes, an average of gains from several altitudes which fall within the performance envelope are used for a specific speed and weight. These gains are computed and placed into tables which *Amelia* loads when a speed command is first given.

3.8 Selection of Feedback Gains for Speed Transition Phase

Table 3.5 contains the poles and zeros for level flight of a B727-200 at 35000 ft. The model is linearized about 280 kt indicated airspeed and idle thrust.

Table 3.5 Typical poles and zeros for a speed transition model.

Poles (sec^{-1})	Transmission zeros for γ_a (sec^{-1})
-5.75×10^{-3}	-5.77×10^{-3}
-2.37×10^{-4}	0
2.34×10^{-4}	

The most obvious item of interest in the table is the pole in the right-half plane. The unstable pole indicates that the aircraft cannot hold a 280 kt indicated airspeed at idle thrust. This is expected since the model is linearized about a deceleration condition. More significant to the design process are the two pole-zero cancellations. This means that for the purposes of gain selection, the system can be treated as first-order with the open loop-pole essentially at the origin. Because of this, the open-loop transfer function from C_L to γ_a can be written as:

$$H_\gamma(s) = \frac{k_\gamma}{s}. \qquad (20)$$

From this, the closed-loop transfer function is:

$$H_\gamma(s) = \frac{K_\gamma k_\gamma}{s + K_\gamma k_\gamma}. \qquad (21)$$

where K_γ is feedback gain. To set the feedback gain for this system, the design criteria is the desired closed-loop time constant τ_c. For a given time constant, the pole placement formula for the feedback gain is:

$$K_\gamma = \frac{1}{k_\gamma \tau_c}. \qquad (22)$$

As before, this derivation applies to a specific speed and altitude. To generate gains which can be used for a wide range of conditions, the feedback gains are computed at various altitude and speed combinations within the flight envelope of a particular aircraft. These gains are then averaged together to produce a gain which corresponds to the aircraft weight.

4 Control Algorithm

The basic framework for the *Amelia* code is established in Secs. 2 and 3. In this section, the focus of the report shifts to the actual software implementation of the ideas developed in the dynamics and control law discussions. In essence, the object of the code is to drive the aircraft from any state to a commanded state following a trajectory described by various parameters passed into the code. This is accomplished by setting the control parameters T, C_L, δ_s and ϕ_r using the control laws discussed in Sec. 3 and applying these parameters to the equations of motion discussed in Sec. 2. Because of the combination of an arbitrary initial state and the non-linearities in the dynamics, the software must carefully manage the combination of reference solution, control law, gains and control limits used. The details of this process are presented in this section.

4.1 Lateral Control

In Secs. 3.1 and 3.2 a linear control law for the commanded bank angle ϕ_r, is developed (see Fig. 3.1 and Eq. (3.3)); however, to account for non-linearities several modifications must be made. The first modification involves large turns. In *Amelia*, large turns are considered to be those heading changes with magnitudes greater than 180°. In order to specify a large turn, a flag called go_around is passed into the control algorithm. The go_around flag has three possible values, 0, −1 and +1. A 0 signifies that a heading change of less than 180° is desired, a −1 indicates a turn of greater than 180° to the left and a +1 a turn greater than 180° to the right. Once the heading change has been determined, it is multiplied by the gain

K_ψ—see Eq. (7)—to set the commanded bank angle ϕ_r. In commercial flight, the bank angle is usually limited to fairly low values (i.e., usually less than 30°). To incorporate this into the control algorithm, the commanded bank angle is limited by a saturation function.

4.2 Command Modes for the Vertical Control

There are four general command modes in which the *Amelia* vertical control algorithm operates. These modes are:

1. Constant indicated airspeed mode;
2. constant Mach number mode;
3. glide slope capture mode; and
4. missed approach mode.

These modes correspond to different operational environments in the terminal area. When an aircraft enters the ATC Center, it is generally at cruise altitude and its operating speeds are specified in Mach number. As it descends, the aircraft operational limits are expressed in indicated airspeed, so speed commands are generally given in terms of these units at lower altitudes. On final approach the command requirements for glide slope capture differ from the first two modes because the ground position of the aircraft relative to the runway and the ILS beam affect the aircraft vertical trajectory. Finally, the missed approach mode stops the aircraft descent and accelerates at level flight until the aircraft reaches a specific climb out speed.

4.3 The Speed-Altitude Plane

To help describe the concepts used in controlling the vertical plane parameters, a graphical view of the problem is used. This is referred to as the speed-altitude plane. An example of the speed-altitude plane is shown in Fig. 4.2. Plotting trajectories in this format has the advantage of showing the actual state of the aircraft relative to the commanded state, much like a phase-plane plot. In Fig. 4.2, three trajectories are shown which start at Mach 0.80 and altitude of 35000 ft, and end at an indicated airspeed of 210 kt and altitude of 11000 ft. Each trajectory represents a different Mach-CAS descent with each phase of the descent clearly represented by a segment. For example, in the 0.84 / 350 kt descent, the four segments represent the acceleration from Mach 0.80 to 0.84, the constant Mach phase, the constant indicated airspeed phase, and the deceleration from 350 kt down to 210 kt.

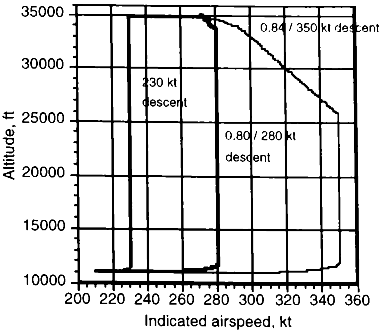


Figure 4.2 Three trajectories in the speed-altitude plane.

The design of the *Amelia* software can be represented in terms of the speed-altitude plane. Such a representation is shown in Fig. 4.3. In this figure, the origin of the plot represents the commanded speed and altitude. The solid lines are the boundaries between different regions defined in the control algorithm. The dashed lines are of analytical interest, but they do not denote major divisions of the software.

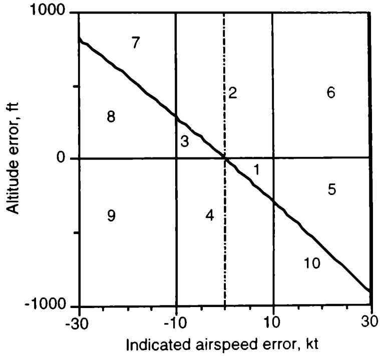


Figure 4.3 Speed-altitude plane regions corresponding to software logic.

Each of the regions bounded by the solid lines and labeled with the numbers represents a different set of control laws, reference solutions, gains and control limits. In physical terms, each set corresponds to a piloting action. Table 4.1 summarizes the default piloting actions modeled in the software for each of the regions shown in Fig. 4.3. These piloting actions are referred to as the default modes. Alternate piloting actions can be specified by passing in flags which are discussed in Sec. 4.8.

In Fig. 4.3, the solid horizontal line divides the speed-altitude plane into regions where the altitude of the aircraft is either too high or too low relative to the commanded altitude. The two vertical lines at −10 kt and +10 kt speed error, split the speed-altitude plane into three regions. Indicated airspeed is shown in this plot, but the same figure can be drawn with Mach number as the horizontal scale.

The middle region is considered to be within the capture range of the speed control law. Conversely, the outside regions employ the speed transition control law in Eq. (10), or a form of it. The diagonal line is the equipotential line. All of the aircraft states lying along this line have the same energy as the commanded states. Energy is defined in terms of altitude potential, and is computed using:

$$E = h + \frac{V_a^2}{2g}. \tag{23}$$

Note that wind velocity is ignored in the definition shown in Eq. (23). For states lying above the equipotential line, above the commanded altitude and outside of the speed capture range, engine throttle is set to idle. For states below the equipotential line, below the commanded altitude and outside of the speed capture range, engine throttle is set to full. For those states which lie between the commanded altitude and the equipotential line and are outside the speed capture range, the thrust is modulated from idle to full throttle, depending upon the aircraft altitude. Within the capture range, engine throttle is determined by how close the aircraft state is to the commanded state.

As mentioned earlier, the piloting actions described in Table 4.1 correspond to a particular combination of reference solution, control law, feedback gains and control limits. Sections 4.4 through 4.7 describe which control functions produce the piloting actions outlined in the table.

5 Conclusions

Amelia is a tool which provides fast affordable simulation of trajectories and radar PVD for multiple aircraft. It is capable of modeling realistic transients produced by maneuvering aircraft. *Amelia* is less expensive to operate than high fidelity cockpit simulators yet is more accurate and realistic than simple point-mass simulations. An early version of *Amelia* is in use at NASA Ames and the FAA real-time simulation laboratories. This same version was also used to successfully remedy the PVD distraction problem in the CTAS simulation.

Table 4.1 Default piloting actions corresponding to regions in Fig. 4.3.

Region	Default piloting action
1	Tweak thrust to be slightly below nominal so aircraft slows down. Capture altitude by adjusting C_L.
2	Descend and capture speed with idle thrust by adjusting C_L. Level off when the aircraft reaches altitude by throttling up.
3	Tweak thrust to be slightly above nominal so aircraft speeds up. Capture altitude with C_L.
4	Climb and capture speed with full throttle by adjusting C_L. Level off when the aircraft reaches altitude by throttling down.
5	Set throttle based on altitude. Climb to commanded altitude.
6	Set throttle to idle. Fly level to lose speed and enter Region 1.
7	Set throttle to idle to lose energy. Descend at a rate which allows the aircraft to accelerate.
8	Set throttle based on altitude. Descend to commanded altitude.
9	Set throttle to full. Fly level to increase speed and enter Region 2.
10	Set throttle to full to increase energy. Climb at a rate which allows the aircraft to decelerate.

References

1. T. Kailath, Linear Systems, Prentice-Hall, Inc., Englewood Cliffs, N.J., 1980, pp. 198-199.
2. R. H. Cannon, Dynamics of Physical Systems, McGraw-Hill Book Company, New York, 1967, pp. 644-682.

AIAA-96-3505-CP

ALGORITHMS FOR AIRCRAFT TRIM ANALYSIS ON GROUND

A. A. Pashilkar[*],
National Aerospace Laboratories, Bangalore 560 017, India

Abstract

In the analysis of aircraft dynamics for up and away flight, the general practice consists of trimming the aircraft for a specific manoeuvre at a specified altitude and mach number. In this paper conditions for quasi-steady equilibrium are motivated for aircraft dynamics in the ground roll phase. Algorithms for implementation of these conditions are also described. By use of these algorithms, the locus of trim points is obtained for a comprehensive take-off manoeuvre of a delta wing aircraft beginning with ground roll, followed by rotation and ending in lift-off. This result is compared with approximate calculations wherein the landing gear states are ignored and linear aerodynamics is assumed. The exact result is also compared with the simulation of the take-off manoeuvre in a pilot in the loop simulator.

Nomenclature

c.g.	: aircraft center of gravity
$C_{L\delta e}, C_{D\delta e}, C_{m\delta e}$	: elevator derivatives w.r.t drag, lift and pitching moment (at cg.)
$D,\ L$	: aerodynamic drag and lift forces
ELS	: Engineer-in-the-Loop-Simulator
F_{nose} , F_{main}	: ground reaction force on the nose and main landing gears
F_{port} , $F_{starboard}$	: ground reaction forces on the port and starboard main landing gears
GFA	: Generic Fighter Aircraft
H_E	: altitude of the aircraft above the ground
k_{eqm}	: equivalent spring constant of the main landing gear (Fig. 6)
l_{no}, l_{mo}	: undeflected lengths of the nose and main landing gears (Fig. 6)
l_m	: deflected length of main landing gears under load R (Fig. 6)
l	: distance between main and nose wheel contact points (Fig. 6)
m	: aircraft mass
$\bar{q}$	: dynamic pressure ($= 1/2\rho V^2$)
R	: normal reaction of main gear (Fig. 4)
P_B, Q_B, R_B	: x, y and z-body axis components of total rotation rate of aircraft
S	: aircraft wing area
T	: total thrust force generated by engine(s)
U_B, V_B, W_B	: x, y and z-body axis components of total velocity of aircraft
V_T	: aircraft total velocity of translation
V_{LOF}	: lift-off speed of aircraft
V_R	: rotation speed of aircraft on ground
W	: aircraft weight
x_t, z_t	: distances of thrust point along x-body and z-body axes (Fig. 5)
x_m, z_m	: dist. of main landing gear contact point in ground axes (Fig. 5)
X_E, Y_E	: position of aircraft along x and y earth fixed axes (on flat earth)

Greek Symbols

α	: angle of attack
β	: angle of sideslip
$\delta_e, \delta_a, \delta_r$	: surface deflections of the elevator, aileron and rudder
δ_{pla}	: power lever angle
θ_{LOF}	: lift-off pitch attitude of aircraft correspd. to lift-off speed V_{LOF}
θ_t	: angle of the thrust line with x-body axis in the xz plane
ϕ, θ, ψ	: euler angles orienting aircraft body axes to earth fixed axes
ρ	: atmospheric density

Introduction

Trim analysis for various aircraft manoeuvres (straight and level, level turn etc.) is a standard and accepted procedure[1]. The author is not aware of similar analysis algorithms for accelerated aircraft manoeuvres on the ground such as take-off. In this paper the trim analysis methods described in reference 1 are generalised to include aircraft dynamics on the ground. This results in an exact calculation of the equilibrium points. Further, it is shown that the general practice of

[*] Scientist, Flight Mechanics & Control Division

linearizing the equations of motion about an equilibrium point for up and away flight can be extended to the ground roll phase.

The algorithms have been implemented on a ground based simulator. In most aircraft design cycles, the flight simulator is brought in at an advanced stage where the usefulness of its results in optimising the design is limited. By introducing the flight simulator in the preliminary design stage and implementing the trim algorithms presented in this paper, the aircraft designer is fed back with more reliable information, which can be used to further optimise his design.

These trim algorithms are also very useful in initialising the aircraft to a deterministic and user specified flight condition on ground for a take-off or landing simulation. This is helpful for validating the landing gear modelling across various platforms against a given template response. Actual flight data could also be used as the template. Previously stored time histories from the same simulator are also sufficient to regenerate the response.

The conditions for quasi-steady equilibrium of aircraft on the ground are derived in Section 1 of this paper. The equilibrium point of this set of equations can be obtained by standard numerical constrained optimisation techniques. The current practice in aircraft industry for take-off analysis makes several simplifying assumptions such as linear aerodynamics and consideration of only the longitudinal degrees of freedom, resulting in simple formulae used in the conceptual design stage. An example of this approach is presented in Section 2 of this paper. As preliminary design progresses and more aerodynamic data becomes available, the designer may wish to refine his earlier calculations. This may not be an easy undertaking, the reason being the high complexity of the aerodynamic database of a typical aircraft. If the algorithm implementation is not general enough, asymmetric operating conditions on take-off (e.g., one engine fail, cross winds etc.) cannot be tackled. Therefore, there is a need for a general trim algorithm for obtaining the flight mechanics parameters of interest, which can work with any database without assumptions.

All the simulations and trim algorithms presented in this paper have been coded on the ELS (Engineer-in-the-Loop Simulator). The ELS at the Flight Mechanics and Controls Division, NAL, is a six degree of freedom, pilot in the loop real time simulation facility. Presently the ELS has been configured to simulate a single engine, tailless delta wing fighter aircraft (henceforth called the GFA).

1. Derivation of Equilibrium Conditions

The rigid body dynamics of aircraft is governed by six degrees of freedom, namely the three translations and three rotations along the spatial co-ordinates[2]. The resulting equations[3] can be referred in standard textbooks. The general aircraft dynamic equations which are non-linear in nature can be cast in the following implicit form[4].

$$\bar{f}(\dot{\bar{X}}, \bar{X}, \bar{U}) = 0 \qquad (1)$$

where $\bar{f}$ is a vector of n scalar non-linear functions f_i. $\bar{X}$ is the state vector $X^T = [V_T \quad \alpha \quad \beta \quad \phi \quad \theta \quad \psi \quad P_B \quad Q_B \quad R_B \quad X_E \quad Y_E \quad H_E]$ and $\bar{U}$ is the input vector $U^T = [\delta_{pla} \quad \delta_e \quad \delta_a \quad \delta_r]$. The equilibrium or singular point(s) of (1) satisfy the following condition.

$$\dot{\bar{X}} = 0 \text{ for some given value of } \bar{U}. \qquad (2)$$

Under the assumptions of constant aircraft mass, flat earth approximation and neglecting atmospheric density effects on aircraft motion, the equations of motion allow us to decouple the earth co-ordinates (X_E, Y_E, H_E) from the closed set represented by (1). Thus, we obtain a set of nine first order differential equations. The state vector is now given by $X^T = [V_T \quad \alpha \quad \beta \quad P_B \quad Q_B \quad R_B \quad \phi \quad \theta \quad \psi]$. For purposes of aircraft performance, stability analysis and control law design, the aircraft motions need to be analysed for various manoeuvres (straight level flight, turning flight, pull-up , push over etc.). To obtain the commonly used equilibrium points for flight mechanics analysis one has to further reduce the state vector from nine to six, corresponding to aircraft degrees of freedom ($X^T = [V_T \quad \alpha \quad \beta \quad P_B \quad Q_B \quad R_B]$) with constraints on either the euler angles (ϕ, θ, ψ) and/or their rates. Therefore (2) implies the following conditions to be satisfied for equilibrium.

$$\dot{V}_T, \dot{\alpha}, \dot{\beta}, \dot{P}_B, \dot{Q}_B, \dot{R}_B \equiv 0, \bar{U} \equiv \text{constant} \qquad (3)$$

subject to appropriate the constraints. The constraints applied depend upon the type of flight mode required[1] (straight and level, level turn, pull up etc.).

Based on (3), one can derive equilibrium conditions for the various phases of the take-off manoeuvre (Fig. 1a). These will be discussed in the following sections.

1.1 Phases of Take-off Manoeuvre

For deriving the trim point conditions the manoeuvre has been divided into three phases as shown in Figure 1b. They are - the accelerated ground roll, the nose wheel lift-off point and rotation to lift-off pitch attitude. In what follows, these phases are examined separately and appropriate trim conditions defined.

a. Accelerated ground roll with thrust set to fixed value (usually at max. dry) : Since the aircraft is accelerating on the runway, all the six accelerations terms in (3) are non zero. A reasonable estimate of the equilibrium state of the aircraft is obtained if the V_T equation is ignored. For this phase of flight all body axis angular rates are close to zero. Further, the bank angle (ϕ) and the yaw angle (ψ) are also equal to zero. Thus, we have the following conditions to be satisfied at trim point.

$$\dot{\alpha}, \dot{\beta}, \dot{P}_B, \dot{Q}_B, \dot{R}_B \equiv 0 \qquad (4a)$$

$$\phi, \phi, \dot{\psi}, \theta \equiv 0 \text{ and } \theta = \alpha \qquad (4b)$$

The case of asymmetries arising out of single engine failure where above assumptions are violated, is examined in the concluding part of Section 1.

Equations (4a,b) are in the form of a constrained minimisation problem with equality constraints. The Newton-Raphson algorithm is suitable for this problem. The following is a commonly used cost function.

$$C = \dot{\alpha}^2 + \dot{\beta}^2 + \dot{P}_B^2 + \dot{Q}_B^2 + \dot{R}_B^2 \qquad (5)$$

In order to satisfy each of the equations in (4a), we need to select the five control variables. The choice of these variables should be such that each equation in (4a) is strongly influenced by at least one different variable. At the same time these control variables must ensure a unique solution. To select such variables we appeal to the physics of the problem. In up and away flight, the usual control variables[1] are $\alpha, \beta, \delta_e, \delta_a, \delta_r$. It can be argued that the $\dot{\alpha}$ equation is a strong function of α , the $\dot{\beta}$ equation that of β , the $\dot{Q}_B$ that of δ_e, $\dot{P}_B$ that of δ_a and the $\dot{R}_B$ equation is a strong function of δ_r. In the case of aircraft resting on both wheels, α depends on θ (constraint (4b)), which in turn depends on the force and moment balance at equilibrium (see Fig. 2). Also in most cases δ_e, the elevator angle is preset during ground roll. Therefore, we need to replace α and δ_e with some other appropriate control variables. The most obvious choice is the forces on the nose (Fnose) and main (Fmain) wheels respectively making $F_{main}, \beta, F_{nose}, \delta_a, \delta_r$ as the set of control variables. Thus we have to satisfy (4a) subject to constraints (4b), for fixed values of the mach number, elevator setting δ_e. and throttle setting δ_{pla}.

When the aircraft reaches rotation speed V_R, the pilot applies back stick (up elevator) to ease the aircraft nose up to the desired take-off attitude. Between the time he starts rotation and the point at which nose wheel lifts off the ground, a trim condition can be defined which satisfies (4). Therefore, we can use the trim algorithm described above to obtain the locus of such trim points as a function of the stick deflection (or equivalently the elevator deflection).

b. Rotation on both wheels till nose-wheel lift-off point : For determination of the exact point of nose wheel lift-off, further conditions need to be satisfied. These conditions are as follows.

i. Ground reaction force on nose wheel is zero (Fnose = 0) (6a)

ii. Nose tire is just touching the ground (in undeflected state). (6b)

The two constraints in (6) have to be imposed in addition to those specified in (4b). Since Fnose is now identically zero, elevator deflection can be taken as a control variable. Thus for nose wheel-off trim, the condition (4a) has to be satisfied subject to the constraints specified in (4b) and (6). The control variables are now $F_{main}, \beta, \delta_e, \delta_a, \delta_r$. For a fixed mach number and throttle setting δ_{pla} the user can determine the minimum elevator required to just lift nose wheel off the ground. In aircraft preliminary design, the designer may want to determine the rotation speed for a given maximum elevator deflection. In such a case, one can replace elevator deflection with mach number as a control variable. Then for a given maximum elevator deflection the user can determine the minimum speed at which rotation can be initiated.

c. Roll on the ground at fixed attitude with nose wheel off : For the case of nose wheel off the ground, the constraints are same as in (4b) and the equations to be minimised are as in (4a). Since the nose wheel is off the ground, the control variables chosen are $F_{main}, \beta, \delta_e, \delta_a, \delta_r$.

In case of asymmetries arising out of engine or single control surface failures, the aileron deflection δ_a can be set to a fixed value, while F_{main} can be split into two forces (one each for either of the main gear struts) F_{port} and $F_{starboard}$. Also the bank angle (ϕ) must be reset every cycle depending on the values of the three landing gear forces F_{nose}, F_{port} and $F_{starboard}$. This concludes the derivation of the trim algorithms for the complete ground roll phase. In the next section approximate equations for trim on ground are derived.

2 Approximate Equations for Aircraft Trim on the Ground

The approximate equations for aircraft trim on the ground can be derived by considering the simplified pitch plane equations only (Fig. 3). The force and moment balance equations are as follows.

Horizontal force balance

$$T\cos(\theta - \theta_t) - D - \mu R = m\ddot{x} \qquad (7)$$

Vertical force balance

$$T\sin(\theta - \theta_t) + L + R = W \qquad (8)$$

Pitching moment balance

$$T(z_t \cos\theta_t - x_t \sin\theta_t) + M - R(x_m + \mu z_m) = 0 \qquad (9)$$

The lift, drag and pitching moment terms can be expanded using the aerodynamic derivative formulation. There are three equations and an equal number of unknowns (θ, R and δ_e) provided forward speed, forward acceleration and the thrust are fixed. Unfortunately, (7) above is poorly conditioned with respect to the pitch attitude θ. The remaining two equations (8) and (9) can be solved simultaneously for R and δ_e, provided θ is known. Using (8) and (9), an estimate of the elevator required to hold a given pitch attitude while rolling on the ground at a given speed and thrust setting can be computed. This computation is sensitive to the values of aerodynamic derivatives used, (particularly $C_{m\delta_e}$) as well as the distances (x_m and z_m).

In order to calculate the minimum elevator required to just lift the nose wheel off the ground, one more relation between the aircraft attitude and R or δ_e is needed. This can be obtained from the geometry in Figure 4.

$$\theta = \tan^{-1}\left(\frac{l_{no} - l_m}{l}\right) \qquad (10)$$

If the strut deflection vs. force characteristics is linear over the range of interest, l_m is related to l_{mo} and normal reaction force R in a direct way as follows.

$$l_m = l_{mo} - \frac{R}{2k_{eqm}} \qquad (11)$$

The factor of two in equation (11) accounts for the presence of two main gears on the aircraft. The minimum elevator required to lift the nose wheel off the ground, main wheel reaction force and pitch attitude at nose wheel lift-off point can now be found using equations (8), (9) (10) and (11) after a few iterations. The results of this method are compared with the exact values in Section 4. In the next section implementation issues for the exact formulation of Section 1 are discussed.

3. Implementation of the Trim Algorithm

The trim constraints explained in section 1 can be implemented with minor modifications to the up and away trim algorithms[1]. For aircraft on ground, the force and moment contributions arise from the following sources - aerodynamic forces and moments, propulsive forces and moments, gravitational forces and ground reaction forces and moments (due to undercarriages). The contributions from aerodynamics, propulsion, and gravitational forces and moments are already computed in the up and away trim algorithms. The flowchart for calculating the forces and moments due to the landing gear model in the aircraft body axis is presented in Figure 5. This portion of the code is common to all the algorithms described above. Hence it can be coded as a separate module.

4. Comparison of Results

The values obtained by the approximate method outlined in Section 2 are compared with the 'exact' values computed by the corresponding trim algorithm described in Section 1.1c in Table 1 for the GFA. It is seen that for all the values in Table 1, the error is within 1deg.

The results of the computations of the minimum elevator point using equations of Section 2 are compared with those obtained from the trim algorithm described in Section 1.1b in Table 2 for various configurations of the GFA. Again the agreement is good for the predicted elevator deflection (within 1.5deg) and the pitch attitude at nose wheel lift-off (within 1deg). The normal reaction force is predicted with a fair degree of accuracy (within 6% of total weight). This is due to the approximation involved in representing the main landing gear by an equivalent spring (Fig. 4).

The results of the trim algorithm for the GFA are plotted in Figure 6 in the pitch attitude vs. elevator deflection plane for various speeds of rotation (200kmph, 220kmph and 240kmph). The initial decreasing segment of each curve corresponds to accelerated ground roll on all wheels. The lowest point of each curve corresponds to the nose wheel lift off point. The increasing segment of each curve corresponds to the rotation upto θ_{LOF} (which for this aircraft is 13deg) after nose wheel lift-off. The maximum up elevator deflection required reflects the static stability (for a given speed) that must be overcome in order to lift the nose wheel off the ground.

In Figure 8 a typical simulation result of a take-off run is presented. The GFA is actually unstable in pitch in the low mach number range. It has been stabilised by the use of pitch rate and normal acceleration feedback. It is clear that the nose wheel (EV04) leaves the ground just as the elevator crosses the minimum elevator required point. The rotation was initiated by the pilot at about 210kmph with a back stick (PSTICK) of roughly 75% of full back stick deflection (42mm). The plots in the second column of Figure 8 show that take-off occurred at about 10sec (EV03).

In Figure 7, the results of the simulation (as presented in Figure 8) are plotted with the results of trim analysis (Fig. 6) in the θ vs. elevator deflection δ_e plane. The simulation results are quite close to the trim calculations, since, the pilot is attempting to execute a smooth rotation manoeuvre and therefore, he needs to apply only an incremental elevator deflection over and above that which is already required to overcome the static stability (Fig. 6). For a non-zero pitch rate, more elevator deflection over and above that shown in the trim plots of Figure 6 is required. Exactly how much more will obviously depend upon the control effectiveness of the aircraft in pitch and its inertia.

Conclusions

An extension of the equilibrium trim analysis to the ground roll phase has been proposed. Its applicability has been demonstrated by comparison with six degree of freedom simulation of a typical fighter aircraft. It is argued that due to the nature of the take-off rotation manoeuvre, the proposed trim strategy is equally applicable to all class of aircraft and for all types of control mechanisms (conventional or fly-by-wire). The comparison also shows that linearisation of the equations of motion in the take-off phase is meaningful. An approximate method for the computation of the parameters of interest to the aircraft design engineer is also presented. This method can be used in the preliminary design stage and gives reasonably accurate results. The advantages of the approximate method is that it relies on a minimum of information about the aircraft and landing gear parameters. It is noted that a simulator platform is the ideal environment for the analysis and application of the trim algorithms presented in this paper.

Acknowledgements

The author is grateful to Wg. Cdr. B. B. Misra of the Indian Air Force for interesting discussions. Thanks are due to Mrs. Padma Madhuranath of the Flight Mechanics & Control Division, National Aerospace Laboratories and Dr. Pradeep of the Indian Institute of Science for their comments.

References

[1] Antoniewicz, R.F.; Duke, E.L.; and Patterson, B.P.: "User's Manual for Interactive LINEAR, a FORTRAN Program to Derive Linear Aircraft Models". NASA TP 2835, 1988.

[2] Etkin, B.: "Dynamics of Flight, Stability and Control', John Wiley and Sons, Inc., Chapt. 4, 1959, pp94.

[3] Padma Madhuranath; and Alok Khare: "CLASS-Closed Loop Aircraft Flight Simulation Software", NAL PD FC 9207, October 1992, pp 23.

[4] Stevens, B.L.; and Lewis, F.L.: "Aircraft Control and Simulation", John Wiley and Sons, Inc., Chapt. 2, Sec 2.5., 1992, pp90

Table 1. Exact and Predicted Elevator as Function of Pitch Attitude

Speed = 200Kmph				Speed = 220Kmph				Speed = 240Kmph			
θ	δe	δe^*	error	θ	δe	δe^*	error	θ	δe	δe^*	error
1.00	0	-	-	0.99	0	-	-	0.94	0	-	-
1.57	-5	-	-	1.67	-5	-	-	1.82	-5	-	-
2.50	-10	-	-	3.08	-10	-	-	3.26	-10	-	-
3.40	-15	-	-	3.48	-15	-	-	3.50	-13.6	-13.9	-0.3
3.60	-20.0	-	-	3.56	-16.6	-16.7	-0.1	5.00	-11.4	-11.4	+0.0
3.62	-20.6	-20.3	+0.3	5.0	-14.3	-14.0	+0.3	10.0	-5.7	-5.7	+0.0
5.0	-18.1	-17.4	+0.7	10.0	-7.0	-7.4	-0.4	13.0	-4.4	-3.9	+0.5
10.0	-9.6	-9.5	+0.1	15.0	-4.7	-4.0	+0.7				
15.0	-5.7	-5.2	+0.5								

Table 2. Exact and Predicted Parameters at Nose Wheel Lift-off Point

CONFIGURATION	Pitch attitude at nose wheel lift-off (deg)		Elevator deflection at nose wheel lift-off (deg)		Main-wheel reaction force at nose wheel lift-off (% of total weight)	
	θ	θ^*	δe	δe^*	R	R^*
CONFIG 1 (200kmph)	3.80	3.06	-18.6	-19.6	55.1	58.0
CONFIG 1 (220kmph)	3.74	3.02	-14.9	-16.0	53.3	57.0
CONFIG 1 (240kmph)	3.67	2.98	-12.1	-13.3	51.3	56.1
CONFIG 2 (200kmph)	3.74	3.02	-17.6	-18.8	54.7	57.7
CONFIG 2 (220kmph)	3.68	2.99	-14.1	-15.4	52.9	56.8
CONFIG 2 (240kmph)	3.62	2.95	-11.5	-12.8	50.9	55.9
CONFIG 3 (200kmph)	3.62	3.03	-20.6	-21.7	58.3	61.0
CONFIG 3 (220kmph)	3.56	2.99	-16.6	-17.8	56.3	60.0
CONFIG 3 (240kmph)	3.50	2.95	-13.6	-14.8	54.0	58.9

*Indicates the values estimated from approximate equations of section 2.

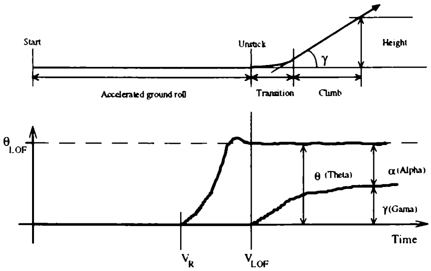


Figure 1a. Typical Pitch Attitude and Flight Path Angle Time Histories for Take-off

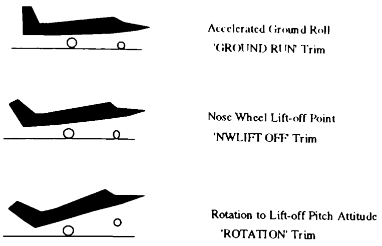


Figure 1b. The Different Phases of a Take-off Manoeuvre

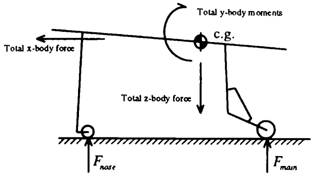


Figure 2. Longitudinal Forces and Moments on the Aircraft on Ground

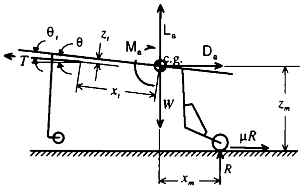


Figure 3. Longitudinal Force and Moment Balance (nose wheel-off)

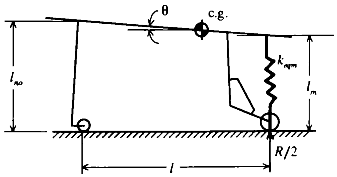


Figure 4. Equivalent Spring Representation of Main Landing Gear

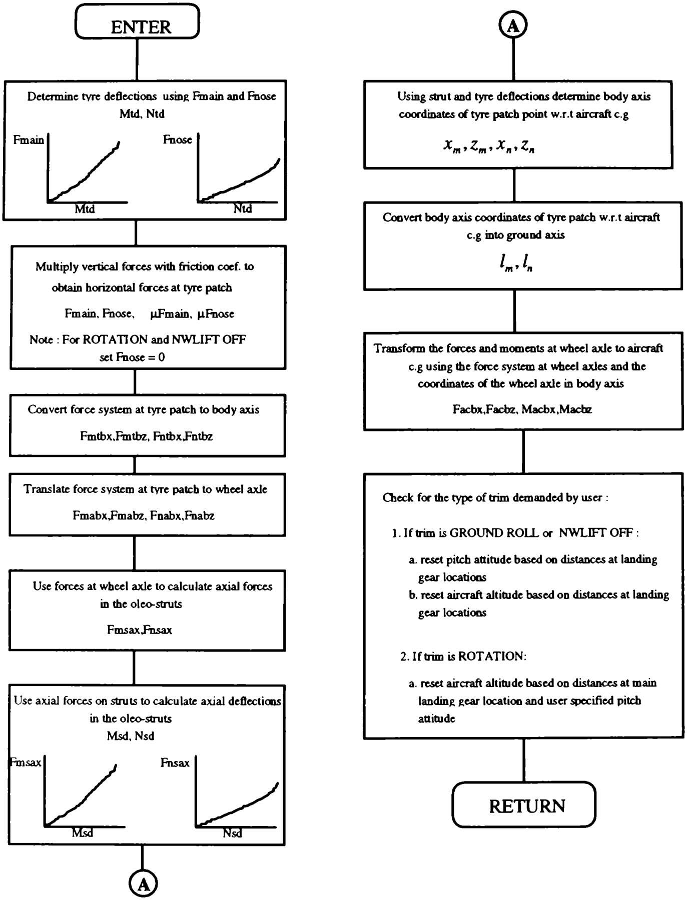

ENTER
Determine tyre deflections using Fmain and Fnose
Mtd, Ntd
Fmain
Mtd
Fnose
Ntd
Multiply vertical forces with friction coef. to
obtain horizontal forces at tyre patch
Fmain, Fnose, μFmain, μFnose
Note : For ROTATION and NWLIFT OFF
set Fnose = 0
Convert force system at tyre patch to body axis
Fmtbx,Fmtbz, Fntbx,Fntbz
Translate force system at tyre patch to wheel axle
Fmabx,Fmabz, Fnabx,Fnabz
Use forces at wheel axle to calculate axial forces
in the oleo-struts
Fmsax,Fnsax
Use axial forces on struts to calculate axial deflections
in the oleo-struts
Msd, Nsd
Fmsax
Msd
Fnsax
Nsd
A
A
Using strut and tyre deflections determine body axis
coordinates of tyre patch point w.r.t aircraft c.g
x_m, z_m, x_n, z_n
Convert body axis coordinates of tyre patch w.r.t aircraft
c.g into ground axis
l_m, l_n
Transform the forces and moments at wheel axle to aircraft
c.g using the force system at wheel axles and the
coordinates of the wheel axle in body axis
Facbx,Facbz, Macbx,Macbz
Check for the type of trim demanded by user :
1. If trim is GROUND ROLL or NWLIFT OFF :
a. reset pitch attitude based on distances at landing
gear locations
b. reset aircraft altitude based on distances at landing
gear locations
2. If trim is ROTATION:
a. reset aircraft altitude based on distances at main
landing gear location and user specified pitch
attitude
RETURN

Figure 5. Flow Chart of Trim Constraints Routine

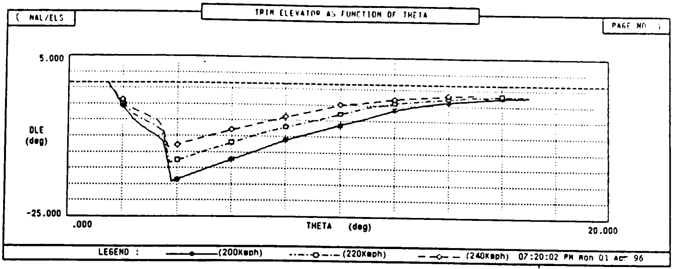


Figure 6. Trim Elevator Vs. Pitch Attitude for Various Speeds

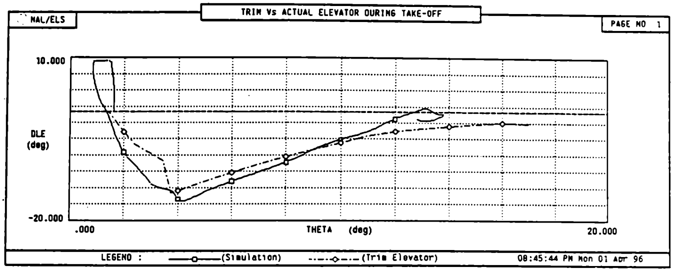


Figure 7. Trim Elevator Required Against Actual Deflection in Simulation

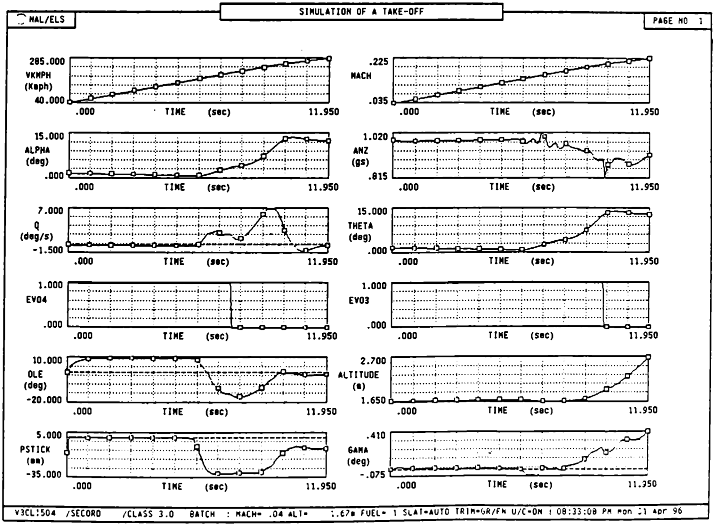


Figure 8. Real Time Simulation Responses for a Take-off

AIAA-96-3506-CP

A PHYSICALLY REPRESENTATIVE AIRCRAFT LANDING GEAR MODEL FOR REAL-TIME SIMULATION

Brent W. York* and Omeed Alaverdi†

Science Applications International Corporation, California, Maryland 20619

Abstract

A detailed aircraft landing gear model developed by the authors is presented. The advantages of this model are numerous. Its accuracy is sufficient for engineering analysis, yet it is applicable to real-time applications. The model is able to represent the complex ground reaction behavior required by such conditions as damaged runways, moving landing surfaces, high surface winds, and large asymmetric loads. The various subsystems are modeled in an intuitive manner, facilitating identification and modification of any physical property by the user.

Problems encountered in the course of developing the model are discussed and solutions to these problems are identified. The model has been incorporated into several simulator systems including a high-fidelity F/A-18A-D simulation, and its observed efficacy in this installation is examined. Future plans for enhancement and expansion of the landing gear model are also discussed.

Introduction

To enable the operation of flight simulators for takeoff and landing maneuvers as well as typical ground operations, it is necessary to model the landing gear system. Historically, aircraft landing gear modeling has been necessarily limited to the specific task for which the simulator is destined. Piloted simulators used for flight training generally lack the sophistication required of a landing gear model which is needed for ground-handling studies. On the other hand, detailed landing gear dynamic models that are used to analyze ground handling and acceptability of the landing gear system are typically too complex for use in piloted simulators when only a small fraction of time is spent on the ground during operation.

The advent of more powerful, less expensive computing systems has enabled more and more complex models to be run in real time, encouraging a convergence between engineering and training simulators. A good example of this trend is the development of the Controls Analysis and Simulation Test Loop Environment (CASTLE) by the Manned Flight Simulator facility at the Naval Air Warfare Center, Aircraft Division.[1] Use of the CASTLE shell facilitates the use of complex, high fidelity simulation models in real time with a pilot in the loop. The same models are also employed to perform various off-line engineering analyses.

Notwithstanding this trend, the majority of real-time simulations in use today still include landing gear models of limited suitability for investigating such problems as landing and takeoff with asymmetric loads or non-conventional surface operations. Simulations such as those used at the Manned Flight Simulator are often called upon to provide just such types of analysis with a pilot in the loop as well as in off-line sessions. These simulations require an aircraft landing gear model which accurately represents landing gear behavior in real time while resorting to the use of as few approximations as possible in order to ensure suitability for meaningful analysis. The landing gear model developed by the authors was designed to meet this requirement.

Specification

A landing gear model intended for use in analytical work must meet certain minimum criteria. The model is required to behave appropriately for all typical ground reaction maneuvers experienced on airfields as well as those required for shipboard operation. Typical ground handling simulation requires modeling for taxiing at both very low and very high speeds and for takeoff and landing operations. The model should be able to simulate steering operation on the ground, braking from "rotors tight" all the way to the anti-skid torque level, and sliding both in the forward and lateral directions. Shipboard operation requires the addition of pitch, roll, heave, and yaw motion to the landing surface and possibly the use of a launch bar and a tailhook, depending on the aircraft.

* Aerospace Engineer, Member AIAA

† Aerospace Engineer, Senior Member AIAA.

The model must have adequate fidelity to represent the landing gear of fixed wing aircraft accurately at high descent rates in order to model landings aboard aircraft carriers correctly. For a helicopter or tilt-rotor aircraft, the model should be able to predict the landing gear system response accurately during very slow vertical descent to landing.

In addition, the ability to represent complex ground reaction behavior such as damaged runways, large asymmetric loads, or high surface winds is desired. The model must be robust enough to avoid anomalous behavior in such extreme conditions, and must provide response that is as accurate as possible within the limits of digital simulation.

A particularly desirable trait for a landing gear model is that it be physically representative. The majority of existing real-time landing gear models use simplified or equivalent models of the landing gear system. In these cases, the impact of any parameter adjustment is not always clear, and often results in undesirable side effects. Ideally, all parameters used to define the behavior of parts of the model should be physically meaningful quantities whose impact on the model is easily understood. For example, adjusting the tire pressure should impact the tire force versus deflection property in a predictable way. Further, a physically representative model is required for parametric analysis of a system.

To avoid "reinventing the wheel," it is of great advantage to design the model from the standpoint of reusability. All subsystems should have well-defined inputs and outputs, so a replacement subsystem can be dropped in if the landing gear model is to be installed on a different aircraft simulation, and the standard subsystem model is unable to represent the new aircraft's components in a satisfactory manner. All standard subsystem models should have parameters that are easily understood for use in tuning the model.

Finally, since the landing gear model will find application in real-time simulation, it is important that the model be as efficient as possible. This stipulation precludes running the entire simulation at hundreds or thousands of cycles per second to obtain reasonable model behavior, and requires that frequency-independent guarantees of stability be included in the model.

Potential Pitfalls

An aircraft landing gear assembly represents a complex system that contains much faster dynamics than the aircraft as a whole in each of its several axes of motion. Some higher frequency dynamics associated with the landing gear system do not impact the overall operation of the simulation and are usually not observable by the pilot or onboard instrumentation. On the other hand, it is necessary to model some of the physics associated with these fast dynamics in order to have a realistic model throughout the operational range of the landing gear. Some of these issues are addressed below.

Discrete Time

The first major concern is one inherent to digital simulation, the fact that time is discontinuous. Given the typical update rate of high fidelity flight simulation models, this issue can largely be ignored in the creation of other simulation components such as aerodynamics or engine models. The landing gear model however, poses special challenges. Since it is not practical to significantly increase the update rate of the entire simulation, it is necessary to identify the dynamic states of the landing gear and isolate those which contribute significantly to the overall behavior of the simulation. The landing gear model must be designed to allow for proper update of these states on a digital computer.

Boundary Conditions

Another potential problem in the development of a landing gear model is the presence of discretized boundary conditions. In the case of the aircraft landing gear, the exact inertial location of the landing gear relative to the landing surface is required to compute the correct reaction force. In a digital simulation, this relative position is updated at regular intervals. Under certain conditions, the change in the relative location of a wheel and the landing surface in a single iteration of the simulation may cause the physical limits of the landing gear assembly to be exceeded. This does not necessarily imply a crash condition.

For example, during an aircraft carrier landing, a typical high performance fighter aircraft simulation updated at a rate of 60 times per second will compute a closure rate of approximately 8 centimeters per time step. Since the change in wheel position each time the aircraft states are updated is comparable to the maximum nose gear tire deflection, this discrete position change can lead to situations in which the simulation detects no tire deflection one frame, and nearly full deflection the next. Such a deflection produces a position-based, spring-like force which, because of the discontinuity in wheel position, will itself be discontinuous. The reaction force can be unrealistically large, bouncing the aircraft back into the air.

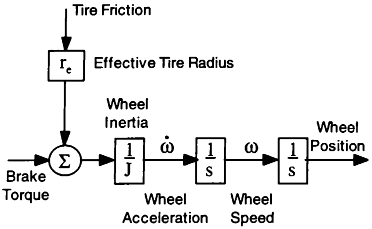


Figure 3. Wheel rotational dynamics.

Although the wheel angular velocity is not directly observable by the pilot, it is needed to properly model the longitudinal motion of the wheel and braking action. In addition, spin-up of the wheel is important in modeling touchdown dynamics.

The rolling torque experienced by a braked wheel is dependent on the slip ratio, defined as:

$$\sigma = \frac{\omega_0 - \omega}{\omega_0}$$

where ω_0 is the angular velocity of the wheel for unbraked rolling, and ω is the wheel angular velocity for braked rolling.[7] When the wheel is rolling at a speed equal to the unbraked nominal speed, the slip ratio is zero and there is no associated rolling torque on the wheel. As the wheel angular velocity is changed, the magnitude of the slip ratio and the corresponding wheel torque magnitude are increased. A slip ratio of magnitude one denotes the point where the wheels are locked. The torques applied to the tire by the ground and the brakes are used to determine the angular acceleration of the wheel each frame, as depicted in Figure 3.

At very low speeds, the slip ratio becomes difficult to calculate due to the digital nature of the system. Therefore, a low speed approximation must be used. This approximation finds slip ratio as a direct function of brake torque, on the assumption that large brake torque will produce a slip ratio of high magnitude, and small brake torque will produce a small slip ratio.

The normalized friction coefficient of the tire is found as a function of the slip ratio. This coefficient is multiplied by the friction characteristic of the landing surface and the tire normal force to obtain the longitudinal force produced at the tire-surface interface.

The wheel model also predicts the side force required to hold the aircraft in a stopped position in order to reject oscillations in the lateral and longitudinal directions. The various wheel objects are required to cooperate to determine how external forces and moments acting on the landing gear model should be divided among the various tires. For example, when faced with a lateral force of 500 pounds due to a crosswind, the wheel objects must decide how much of this force must be opposed by each wheel. This problem is not a simple one, and at the moment has only been coded for the case of tricycle landing gear.

Brake Objects

The baseline landing gear model contains a very simple brake model with a rudimentary anti-skid braking system. Brake pedal inputs from the pilot are converted to a corresponding brake pressure, and an appropriate torque is applied to the axle. The anti-skid braking algorithm monitors slip ratio and limits brake pressure as required to avoid skidding. Various parts of this model may be replaced with more complex objects for braking system analysis tasks. For example, the simplified anti-skid system may be replaced with models for wheel speed sensors, brake pressure sensors, and the actual anti-skid braking algorithm for the aircraft being analyzed.

Strut Objects

A strut object contains only one degree of freedom, the strut deflection. The deflection of the strut is used to determine the location and orientation of the axle, which in the most general sense has four degrees of freedom: three translational and one rotational (that is, tilt or "roll"). The actual number of degrees of freedom that the axle has is dependent on the number of struts contained in the landing gear assembly.

A typical strut assembly such as that found in the main landing gear of the F/A-18C (Figure 4) contains a single strut and one or more rigid members. The deflection of the single strut is translated in real-time into the location and tilt of the axle through the use of

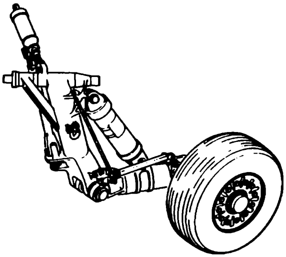


Figure 4. F/A-18C main landing gear assembly.

pre-computed lookup tables. A linear function interpolation is used in obtaining these values for the sake of efficiency. In principle, any number of struts and rigid members can be modeled in each landing gear assembly, as appropriate.

Steering Objects

The steering model introduces another degree of freedom to the axle by allowing yawing motion. This object is generally used to represent steering of the nosewheel in aircraft with conventional landing gear. The basic model includes a simple actuator model for pilot-commanded steering as well as nosewheel castoring.[9]

Surface Geometry Objects

The landing surface model in the simplest sense represents a flat plate on which the aircraft can land. To this simple concept are added several enhancements, the most important of which is the ability to allow the plate to move in real-time. This motion capability is necessary to provide simulation of shipboard operations, with an aircraft carrier deck possibly pitching in heavy seas. In addition, refinements such as differing surface orientations and compositions, damaged surfaces, and structures are possible, features which are important for analysis of aircraft-surface interaction.[10]

Launch Bar and Tailhook Objects

The landing gear model contains provisions for fixed-wing aircraft operations from an aircraft carrier in the form of a launch bar object and a tailhook object. These objects are meant to interact with an external ship model containing catapults and arresting gear.

Force and Moment Summation

An executive routine is provided with the landing gear model. It is the responsibility of this routine to ensure that all the landing gear objects are invoked properly and at the correct update rate. The executive transfers the forces and moments generated by the various objects to the proper coordinate frame and reference locations and sums them. The resulting total ground reaction forces and moments are passed to the aircraft equations of motion module.

Installation and Evaluation

The landing gear model was first installed in a high-fidelity simulation of an F/A-18C aircraft used in a pilot-in-the-loop simulator. Pilots experienced in flying the actual aircraft were asked to perform various operations in the simulator involving the landing gear. Takeoffs and landings from airfields as well as moving aircraft carriers were performed, both with and without crosswinds. Simple taxi tasks such as minimum radius turns and navigation from the ramp to the active runway were also carried out. The pilots' comments about the handling qualities of the aircraft were noted and used to evaluate the landing gear model. On the whole, pilots have been pleased with the response of the model, describing it as more representative of the behavior of the actual aircraft than the previously employed simplified real-time model.

Successful installations have also been performed in other high-fidelity simulations, from large fixed-wing cargo aircraft to tilt-rotor aircraft. The model has received favorable pilot comments during evaluation of these installations as well. However, no rigorous validation of the model has been performed for any aircraft. This omission is due to the lack of available detailed flight test data for aircraft operating on or near a landing surface.

Future Work

The landing gear model presented here was designed to allow for incremental or program-specific improvements to the baseline code. Work is ongoing to improve the accuracy of the landing gear model by evaluating the effects of the empirical equations used in the tire and strut models and determining any possible deficiencies. Other specific enhancements and improvements planned for the landing gear model include better distribution of reaction force between various wheel objects and more detailed surface modeling.

Validation against actual aircraft data is an important goal. The quest to obtain accurate and complete data for use in validating the model has yet to yield results. The ideal solution would be to instrument an aircraft specifically for landing gear model validation and to specify the data set and maneuvers required.

Conclusion

Landing gear models intended for use in real-time simulation are not as detailed as those employed solely in off-line engineering analysis. The model outlined here represents an attempt to merge the two types of simulation models to obtain the best of both worlds. This model shows promise for use in the next generation of real-time flight simulators in support of flight test efforts, pilot familiarization, envelope expansion, and accident investigation. Once a rigorous validation effort has been completed, confidence in the accuracy of the model will be confirmed.

References

1. Nichols, James, "The Generic Simulation Executive at Manned Flight Simulator," AIAA 94-3429, AIAA Flight Simulation Technologies Conference, Scottsdale, AZ, August 1994.

2. McHehee, J.R., and Carden, H.D., "A Mathematical Model of An Active Control Landing Gear for Load Control During Impact and Roll-Out," NASA TN D-8080, National Aeronautics and Space Administration, Langley Research Center, Hampton, VA, February 1976.

3. Skorupa, J.A., Captain, USAF, "System Simulation in Aircraft Landing Gear and Tire Development," Thesis Presented to the Faculty of the School of Engineering of the Air Force Institute of Technology Air University, Wright Patterson AFB, Ohio, December 1976.

4. Clark, J.W., Jr., "CTOL Ski Jump Dynamic Analysis Model and Computer Program," NADC-83035-60, Naval Air Development Center, Warminster, PA, June 1983.

5. McFarland, R.E., "Anticipation of the Landing Shock Phenomenon in Flight Simulation," NASA Technical Memorandum 89465, National Aeronautics and Space Administration, Ames Research Center, Moffett Field, CA, September 1987.

6. Haraldsdottir, A. and Howe, R.M., "Multiple Frame-Rate Integration," AIAA 88-4579, AIAA Flight Simulation Technologies Conference, Atlanta GA, September 1988.

7. Smiley, R.F. and Horne, B., "Mechanical Properties of Pneumatic Tires with Special Reference to Modern Aircraft Tires," NASA Technical Report R-64, National Aeronautics and Space Administration, Langley Research Center, Langley Field, VA, 1960.

8. Tanner, J.A. and Dreher, R.C., "Cornering Characteristics of a 40x14-16 Type VII Aircraft Tire and a Comparison with Characteristics of a C40x14-21 Cantilever Aircraft Tire," NASA Technical Note TN-D-7351, National Aeronautics and Space Administration, Langley Research Center, Hampton, VA, October 1973.

9. Daugherty, R.H. and Stubbs, S.M., "A Study of the Cornering Forces Generated by Aircraft Tires on a Tilted, Free-Swiveling Nose Gear," NASA Technical Paper 2481, National Aeronautics and Space Administration, Langley Research Center, Hampton, VA, October 1985.

10. Cox, J.J., Jr. Lt. Col., Henghold, W.M., Lt. Col., Russell, J.J., Maj. USAF, "A Literature Search & Review of the Dynamics of Aircraft-Surface Interaction," Department of Civil Engineering, Engineering Mechanics and Materials, United States Air Force Academy, Colorado, June 1979.

AIAA-96-3507-CP

FROM TENSOR MODELING TO MATRIX COMPUTER CODE: EFFICIENT FLIGHT SIMULATION DEVELOPMENT

Peter H Zipfel
USAF ASC/XRWD
Eglin AFB, FL 32542

Abstract

The development of flight simulations is burdened by a multitude of coordinate systems whose transformations vary with time. A tensor modeling technique is reviewed that formulates the equations of motion in a form that is invariant under time dependent coordinate transformations. For programming, coordinate systems are selected and the tensor equations convert into matrix relations that can be coded directly. The methodology is exemplified by the 6DOF equations of motion for flat earth, rotating earth, and spinning missiles. Twenty years of applications to flight simulations and of teaching experience at the University of Florida are briefly summarized.

Nomenclature

$[*]$ Tensor

$[\bar{*}]$ Transpose of a tensor

$[*]^{-1}$ Inverse of a tensor

$[T]^{BA}$ Transformation matrix of coordinate system B wrt system A

$[*]^A$ Coordinate matrix of a tensor expressed in the $]^A$ coordinate system

$[s_{BA}]$ Displacement vector (first order tensor) of point B with respect to (wrt) point A

$[v_B{}^A]$ Velocity vector (first order tensor) of point B wrt frame A

$[\omega^{BA}]$ Angular velocity vector (first order tensor) of frame B wrt frame A

$[\Omega^{BA}]$ Skew symmetric form of $[\omega^{BA}]$

$[R^{BA}]$ Rotation tensor (second order tensor) of frame B wrt frame A

$[p^{BA}]$ Linear momentum (first order tensor) of body B wrt frame A

$[l_C{}^{BA}]$ Angular momentum (first order tensor) of body B wrt frame A referred to point C

$[I_C{}^B]$ Intertia tensor (second order tensor) of body B referred to point C

$[D^A *]$ Rotational time derivative wrt frame A

$[d*/dt]^A \equiv [\dot{*}]^A$ Time derivative operating on the vector coordinates of $[*]^A$

m mass (zeroth order tensor or scalar) of body B

wrt "with respect to"

superscripts are frames and *subscripts* are points

Introduction

Those of us who spend weeks and months at-a-time modeling flight dynamics, writing code and debugging subroutines have looked for ways to make our job easier. One of the crosses to bear is the multitude of coordinate systems that are needed to describe the motions of a flight vehicle. The ANSI/AIAA Standard R-004-1992[1] defines eleven coordinate systems associated with one vehicle alone. In multi-body engagements this number is even larger.

If the transformations between these coordinate systems were time *independent*, the vectors and their time derivatives could easily be expressed in *any* of the coordinates. Yet just about all our coordinate systems move relative to each other. So we have to go through the gymnastics of special time derivatives that are valid only in the particular coordinate system of the vector components and, therefore, we find ourselves writing the equations in component form rather than taking advantage of the matrix coding that Fortran 90 or C++ permit.

In the 70's, when the IBM punch card was still king, we developed a methodology that starts by modeling the kinematic and dynamic equations in a *tensor* format that is *independent* of coordinate systems. After manipulating the equations to make them suitable for the simulation task at hand, the final step is to select the appropriate coordinate systems and program the equations in *matrix* form that are executed either by array subroutines or, more recently, by array intrinsic functions. This approach was refined over two decades and applied to air-to-air, air-to-ground and transatmospheric simulations, both manned and unmanned. Itwas also introduced into the academic world and is currently being taught at the University of Florida Graduate Engineering and Research Center [2].

This paper gives an *overview* of the tensor modeling technique and, with the help of examples, demonstrates the matrix implementations. Our experience with several simulations is summarized.

Tensor Modeling

Newton's Law

Vector mechanics is the mainstay of flight modeling. When time derivatives are formed however, specific coordinate systems must be identified and, as a consequence, the equations lose their invariancy under coordinate transformations. The use of the *rotational time derivative* ^{3} eliminates this problem and allows an invariant formulation of the dynamic equations which are valid in any coordinate system.

As an example, Newton's law in inertial coordinates is

$$m\frac{d\bar{v}}{dt} = \bar{f}$$

but in body axes it has the *different* form

$$m\left(\left.\frac{d\bar{v}}{dt}\right|_{body} + \bar{\omega}\times\bar{v}\right) = \bar{f}.$$

Introducing the rotational time derivative D^I with respect to the inertial frame, Newton's law has the same form in both coordinate systems I and B:

$$m[D^I v]^I = [f]^I \text{ and } m[D^I v]^B = [f]^B$$

The key is the careful distinction between *coordinate systems* and reference *frames*. Since this formulation is the same in any coordinate system it is *invariant* under coordinate transformations and therefore a true tensor equation that may be written without any reference to a coordinate system:

$$m[D^I v] = [f].$$

Conversely, $m\frac{d\bar{v}}{dt} = \bar{f}$ is not valid in any coordinate system. Because of the time derivative, it holds only in an inertial coordinate system. We will now address these issues in detail.

Coordinate Systems

First we will have to deal with a few definitions. Coordinates are *ordered* algebraic numbers (*triples, n-tuples)* and the corresponding coordinate system is an *abstract* entity that establishes an *one-on-one* correspondence between the elements of the Euclidean three-space and the coordinates. Coordinate axes are the *geometrical images* of mathematical scales of algebraic numbers. *Cartesian* coordinate systems are coordinate systems in the Euclidean space for which the Euclidean metric $ds^2 = \sum_i dx_i^2$ holds *also* for finite differences $\Delta s^2 = \sum_i \Delta x_i^2$. This restriction eliminates cylindrical and spherical coordinates.

A *coordinate transformation* is a re-labeling of each element in Euclidean space with new coordinates according to a certain algorithm. A coordinate system is said to be *associated* with a frame if the coordinates of the frame points are time invariant. All coordinate systems embedded in one frame form a *class* $\aleph$. The entity of all classes over all frames are the *allowable* coordinate systems.

The group of allowable coordinate transformations are the transformations between allowable coordinate systems. The elements of these transformations are, in general, time dependent. Only within one class are the transformations time invariant.

We will use only orthogonal, right handed Cartesian coordinate systems. the transformations between these coordinate systems are orthogonal.

Some *examples* may clarify these concepts:

(1) Coordinates are matrices,

$$\text{e.g. ordered triples: } = \begin{bmatrix} 3.1 \\ 2 \\ -1.2 \end{bmatrix}$$

$$\text{ordered 9-tuples:} = \begin{bmatrix} 1.2 & 6.8 & -1 \\ 0 & 1 & 9.1 \\ 5 & 8 & -1 \end{bmatrix}$$

(2) Coordinate systems associate matrices to the Euclidean three-space:

$$\begin{bmatrix} \text{first direction} \\ \text{second direction} \\ \text{third direction} \end{bmatrix} = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix}$$

(3) Coordinate axes determine only units, direction, and sense, they *do not* define an origin, therefore they do not have to originate from the same point.

(4) Coordinate transformations re-lable coordinates:

$$\begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} \xleftarrow{\text{relabelling}} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix};$$

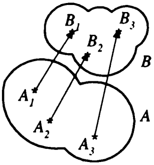
where $[T] = \begin{bmatrix} t_{11} & t_{12} & t_{13} \\ t_{21} & t_{22} & t_{23} \\ t_{31} & t_{32} & t_{33} \end{bmatrix}$ is an example of such a transformation

To identify a particular coordinate system, we affix a superscript to the bracket. For instance, the vector $[f]$, expressed in inertial axes I, is $[f]^I$. If $[T]^{BI}$ is the coordinate transformation of body wrt inertial coordinates then the body coordinates are obtained from the matrix multiplication

$$[f]^B = [T]^{BI}[f]^I.$$

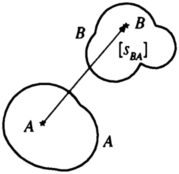
Reference Frames

A Frame is the mathematical representation of an *idealized* physical object whose elements do not move with respect to each other. Examples are rigid bodies, inertial space, air moving at a constant velocity, and reference frames established by an observer.

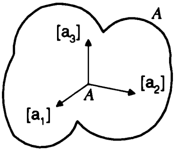


To define the **position** of a frame let A and B be two frames with three non-collinear points A_1, A_2, A_3 and B_1, B_2, B_3, respectively. The position of frame B wrt frame A is determined by three *displacement vectors* $[s_{B_iA_i}]$, $i = 1,2,3$. Six of the nine vector coordinates are independent; i.e., a frame has six degrees of freedom.

The position of a frame can be decomposed into *location* and *orientation*, each representing three degrees of freedom.



To define the **location,** let A and B be points (base points) of frames A and B, respectively. The displacement vector $[s_{BA}]$ determines the *location* of frame B relative to frame A.



Now, we need to define a **triad.** A triad is a set of three orthonormal *base vectors* with magnitudes of one that connect the base point A with three other points of a frame A.

$$[a_1],[a_2],[a_3] \quad \hookleftarrow \; [\bar{a}_i][a_j] = \begin{Bmatrix} 0 \; for \; i \neq j \\ 1 \; for \; i = j \end{Bmatrix}, \begin{Bmatrix} i = 1,2,3 \\ j = 1,2,3 \end{Bmatrix}$$

Among the coordinate systems *associated* with each frame, there is one, the *preferred* coordinate system, that gives the coordinates of the base vectors in the simple form

$$[1 \quad 0 \quad 0], [0 \quad 1 \quad 0], [0 \quad 0 \quad 1].$$

The **orientation** of a frame B wrt frame A is determined by the orientation of the *B-triad* relative to the *A-triad*.

The position of a frame is uniquely determined by its triad, the location by the base point and the orientation by the base vectors.

Rotational Time Derivative

The rotational time derivative is the key to the invariant formulation of dynamics for all *allowable* coordinate systems. Based on the contra-variant time derivative vector, first introduced by Wrede [3], the tensor property of the rotational time derivative of first and second order tensors was proven by Zipfel[4]. Here, we provide a heuristic derivation of the concept.

We begin by defining the nomenclature for linear velocity and acceleration. The *linear velocity* of point B wrt to frame I is $[v_B^I]$, a first order tensor. We may interpret B as the center of mass of a vehicle and I as an inertial reference frame. Its components in the inertial coordinate axes $]^I$ are

$$[\overline{v_B^I}]^I = [u^I \quad v^I \quad w^I]^I.$$

The inertial acceleration of point B wrt inertial frame I is defined in inertial axes:

$$[a_B^I]^I = \left[\frac{dv_B^I}{dt}\right]^I.$$

For two inertial coordinate systems , $]^{I1}$ and $]^{I2}$, *associated* with frame I we have the component relationship

$$[v_B^I]^{I1} = [T]^{I1I2}[v_B^I]^{I2}.$$

Taking the time derivative we get the inertial accelerations expressed in the two inertial coordinate systems:

$$\left[\frac{dv_B^I}{dt}\right]^{I1} = [T]^{I1I2}\left[\frac{dv_B^I}{dt}\right]^{I2};$$

i.e., the inertial acceleration transforms like a first order tensor as long as the coordinate systems are *associated* to the inertial frame (the transformation matrices are time invariant). However, if we consider all *allowable* coordinate axes; e.g., inertial axes $]^I$ and earth axes $]^E$, and take the time derivative of

$$[v_B^I]^I = [T]^{IE}[v_B^I]^E$$

we get

$$\frac{d}{dt}[v_B^I]^I = \frac{d}{dt}\{[T]^{IE}[v_B^I]^E\}$$

$$= [T]^{IE}\left\{\left[\frac{dv_B^I}{dt}\right]^E + [\bar{T}]^{IE}\left[\frac{dT}{dt}\right]^{IE}[v_B^I]^E\right\} \qquad (1)$$

we discover that the regular time derivative applied to a first order tensor does *not* transform like a tensor (the transformation matrix is now time dependent).

Now, define the *rotational time derivative* D^I wrt frame I operating on any first order tensor $[x]$ expressed in coordinate system $]^E$ as:

$$[D^I x]^E \equiv \left[\frac{dx}{dt}\right]^E + [T]^{EI}\overline{\left[\frac{dT}{dt}\right]}^{EI}[x]^E.$$

Let it operate on $[v_B^I]^I$:

$$[D^I v_B^I]^I = \left[\frac{dv_B^I}{dt}\right]^I + [T]^{II}\underbrace{\overline{\left[\frac{dT}{dt}\right]}^{II}}_{=0}[x]^I = \left[\frac{dv_B^I}{dt}\right]^I \qquad (2)$$

which is the left hand side of Eq(1). Furthermore, let it operate on $[v_B^I]^E$:

$$[D^I v_B^I]^E \equiv \left[\frac{dv_B^I}{dt}\right]^E + [T]^{EI}\overline{\left[\frac{dT}{dt}\right]}^{EI}[v_B^I]^E. \qquad (3)$$

Substitute Eqs(2) and (3) into (1) we obtain

$$[D^I v_B^I]^I = [T]^{IE}[D^I v_B^I]^E$$

where we used $[T]^{EI} = [\bar{T}]^{IE}$. Therefore, the rotational derivative D^I of vector $[v_B^I]$ *transforms like a first order tensor.*

Coordinate systems $]^I$ and $]^E$ may be replaced by any pair of *allowable* coordinates and likewise $[v_B^I]$ may stand for any first order tensor $[x]$. Thus, $[D^I v_B^I] = [a_B{}^I]$ is the *inertial acceleration tensor* and $[D^I x]$ is the time rate of change tensor of $[x]$ wrt frame I.

In summary, the rotational time derivative of a *first order tensor* $[x]$ wrt any frame A, $[D^A x]$, and expressed in any allowable coordinate system $]^B$ is:

$$\boxed{[D^A x]^B \equiv \left[\frac{dx}{dt}\right]^B + [T]^{BA}\overline{\left[\frac{dT}{dt}\right]}^{BA}[x]^B} \qquad (4)$$

It transforms like a first order tensor

$$[D^A x]^D = [T]^{DC}[D^A x]^C$$

where $]^C$, $]^D$ are any allowable coordinate systems.

Euler Transformation

We have seen that the *rotational time derivative* is the key to the invariant formulation of time phased dynamic systems. It is a time operator that depends on the reference frame. Sometimes it is desirable to change this reference frame; e.g., from inertial to body frame. *Euler's generalized transformation* governs the change of frame of the rotational time derivative.

Let A and B be two arbitrary frames related by the angular velocity tensor $[\Omega^{BA}]$. Then, for any vector $[x]$, the following transformation of the rotational time derivatives holds:

$$\boxed{[D^A x] = [D^B x] + [\Omega^{BA}][x]} \qquad (5)$$

This theorem is a direct consequence of the *isentropic* property of space[5]; i.e., the rotational time derivative of a vector is invariant under spatial rotations of all points and frames involved. Specifically, if $[R^{BA}]$ is the *rotation tensor* of frame B wrt frame A, then, because of the isentropic property of space, the rotational derivative of $[x]$ wrt frame A, $[D^A x]$, can also be evaluated as follows: (a) first rotate $[x]$ through $[R^{BA}]$ to obtain $[R^{BA}][x]$; (b)take the rotational time derivative wrt the rotated frame, now called B, $\left[D^B\left([R^{BA}][x]\right)\right]$; and (c) rotate the result back through $[R^{BA}]$ into the original orientation:

$$[D^A x] = [\overline{R^{BA}}]\left[D^B\left([R^{BA}][x]\right)\right]$$

The chain rule applied to the right side yields:

$$[D^A x] = [D^B x] + [\overline{R^{BA}}][D^B R^{BA}][x] \qquad (6)$$

If we can show that

$$[\overline{R^{BA}}][D^B R^{BA}] = [\Omega^{BA}]$$

then Eq(5) is proven. To do so we interchange A and B and execute the same three steps:

$$[D^B x] = [D^A x] + [\overline{R^{AB}}][D^A R^{AB}][x] \qquad (7)$$

Adding Eqs (6) and (7) yields:

$$[\overline{R^{BA}}][D^B R^{BA}] = -[\overline{R^{AB}}][D^A R^{AB}]$$

The right hand side is the desired $[\Omega^{BA}]$ because:

$$-[\overline{R^{AB}}][D^A R^{AB}] = -[R^{BA}][\overline{D^A R^{BA}}]$$

$$= -[\overline{D^A R^{BA}][\overline{R^{BA}}}] = -[\overline{\Omega^{BA}}] = [\Omega^{BA}]$$

because[2]

$$[\Omega^{BA}] = [D^A R^{BA}][\overline{R^{BA}}].$$

This completes the proof of the theorem.

A *heuristic* development leads from the classical to the generalized Euler transformation. We start with the textbook transformation of time derivatives

$$\left(\frac{d\bar{x}}{dt}\right)_{Frame\ A} = \left(\frac{d\bar{x}}{dt}\right)_{Frame\ B} + \bar{\omega} \times \bar{x}$$

(which is not an invariant tensor formulation). Re-write it in matrix nomenclature

$$\left[\frac{dx}{dt}\right]^A (=) \left[\frac{dx}{dt}\right]^B + [\Omega^{BA}][x],$$

and do this in consistent coordinates

$$\left[\frac{dx}{dt}\right]^A = [T]^{AB}\left(\left[\frac{dx}{dt}\right]^B + [\Omega^{BA}]^B[x]^B\right).$$

Then we introduce the rotational time derivatives

$$[D^A x]^A = [T]^{AB}\left([D^B x]^B + [\Omega^{BA}]^B[x]^B\right)$$

and obtain the invariant tensor formulation of the Euler transformation

$$[D^A x] = [D^B x] + [\Omega^{BA}][x].$$

Note, the Generalized Euler Transformation is an invariant tensor concept *independent* of any coordinate system.

So far, I have heavily taxed your patience with the basics of tensor modeling. Just remember that coordinate systems are mathematical constructs and frames represent physical things and keep them separated. If you accept the rotational derivative as useful and Euler's transformation as valid you should have no problems with the following derivation of several sets of equations of motion.

Equations of Motion

We shall derive three sets of 6DOF equations of motion from Newton's and Euler's laws. The *flat earth model* is appropriate for short duration flights near the earth surface. The *rotating earth model* must be used for long duration flights or for transatmospheric trajectories. The *spinning missile* model is useful for rapidly rotating bodies.

The equations will be derived in a tensor form that is valid in any coordinate system. Much physical insight can be gleaned from these tensor equations. Yet eventually we must assign numerical values to the variables. To do so we introduce coordinate systems. Then the equations become matrix relationships that can be coded up as array statements and solved on the computer.

Flat Earth Model

We derive the translational equations of a flight vehicle subjected to the aerodynamic and thrust forces $[f_{a,t}]$ and the gravitational acceleration $[g]$. From Newton's law

$$m[D^I v_B^I] = [f_{a,t}] + m[g], \qquad (8)$$

where m is the vehicle mass and $[v_B^I]$ is the velocity of the missile c.m., B, wrt the inertial reference frame I. The flat earth assumption allows us to declare the earth frame E as the inertial frame. Therefore, Newton's law becomes

$$m[D^E v_B^E] = [f_{a,t}] + m[g].$$

To model the aerodynamic forces we need to have the incidence angles available. They are calculated from the velocity vector as perceived from the vehicle as reference frame. Its time rate of change is therefore relative to the vehicle, $[D^B v_B^E]$. This desired shift of reference frame is accomplished through the *Euler transformation*

$$m[D^B v_B^E] + m[\Omega^{BE}][v_B^E] = [f_{a,t}] + m[g] \qquad (9)$$

These are the translational equations of a vehicle over a flat earth. The second term on the left hand side is identified as the *tangential acceleration* term. The equation is valid in any coordinate system.

For 6DOF simulations they are expressed in body axes

$$m[dv_B^E / dt]^B + m[\Omega^{BE}]^B[v_B^E]^B = [f_{a,t}]^B + m[g]^B.$$

The rotational time derivative has become the simple time operator applied to the three velocity components. To complete the *matrix equation* we express the gravity acceleration term in the more convenient local level coordinates using the transformation matrix $[T]^{BL}$ (derived later):

$$m[dv_B^E / dt]^B + m[\Omega^{BE}]^B[v_B^E]^B = [f_{a,t}]^B + m[T]^{BL}[g]^L$$

These equations can be coded directly into a simulation with program languages like Fortran 90 or C. Or the simulation environment (e.g. CADAC[6]) may provide the utility subroutines that enable vector state variable integration and matrix manipulations.

In coordinate form they take on the following form

$$m\left\{\begin{bmatrix}\dot{u}\\ \dot{v}\\ \dot{w}\end{bmatrix}^B + \begin{bmatrix}0 & -r & q\\ r & 0 & -p\\ -q & p & 0\end{bmatrix}^B \begin{bmatrix}u\\ v\\ w\end{bmatrix}^B\right\}$$

$$= \begin{bmatrix}f_{a,t_1}\\ f_{a,t_2}\\ f_{a,t_3}\end{bmatrix}^B + \begin{bmatrix}t_{11} & t_{12} & t_{13}\\ t_{21} & t_{22} & t_{23}\\ t_{31} & t_{32} & t_{33}\end{bmatrix}^{BL} \begin{bmatrix}0\\ 0\\ mg\end{bmatrix}^L$$

The integration of these differential equations yields the velocity vector which must be integrated once more to obtain the location of the missile c.m., B, wrt an earth refence point E

$$[D^E s_{BE}] = [v_B^E]. \qquad (10)$$

The integration is best carried out in the local level coordinate system. Therefore, given $[v_B^E]^B$, we program the second set of differential equations as

$$[ds_{BE} / dt]^L = [\overline{T}]^{BL}[v_B^E]^B$$

and in coordinate form

$$\begin{bmatrix} (ds_{BE}/dt)_1 \\ (ds_{BE}/dt)_2 \\ (ds_{BE}/dt)_3 \end{bmatrix}^L = \overline{\begin{bmatrix} t_{11} & t_{12} & t_{13} \\ t_{21} & t_{22} & t_{23} \\ t_{31} & t_{32} & t_{33} \end{bmatrix}}^{BL} \begin{bmatrix} u \\ v \\ w \end{bmatrix}^B$$

The rotational degrees of freedom are governed by *Euler's law* that states that the time rate of change of angular momentum equals the externally applied moments through the vehicle c.m.,B

$$[D^E l_B^{B\,E}] = [m_B] \qquad (11)$$

where $[l_B^{B\,E}] = [I_B^B][\omega^{BE}]$, is the angular momentum of Body B wrt frame E referred to the c.m., $[I_B^B]$ is the moment of inertia of missile body B referred to the c.m. and $[m_B]$ the aerodynamic and thrust moments referred to the c.m.

Just as in the case of the translational equations, we adopt the viewpoint from the vehicle body frame B and use *Euler's transformation* to transfer the rotational derivative to the body frame

$$[D^B l_B^{B\,E}] + [\Omega^{BE}][l_B^{B\,E}] = [m_B].$$

Let us look at the rotational derivative, expanding the angular momentum vector and applying the chain rule

$$\begin{aligned} [D^B l_B^{B\,E}] &= \left[D^B([I_B^B][\omega^{BE}])\right] \\ &= \left[D^B I_B^B\right][\omega^{BE}] + [I_B^B]\left[D^B\omega^{BE}\right] \\ &= [I_B^B]\left[D^B\omega^{BE}\right] \end{aligned}$$

For a rigid body, $[D^B I_B^B]$ is zero; a simplification that justifies the transformation to the body frame. The equations become now

$$[I_B^B][D^B\omega^{BE}] + [\Omega^{BE}][I_B^B][\omega^{BE}] = [m_B]. \qquad (12)$$

These are the rotational equations of a vehicle with the earth assumed to be the inertial refence frame.

Introduce the body axes B to the coordinate form

$$[I_B^B]^B[d\omega^{BE} / dt]^B + [\Omega^{BE}]^B[I_B^B]^B[\omega^{BE}]^B = [m_B]^B$$

In matrices (body axes are principle axes):

$$\begin{bmatrix} I_{11} & 0 & 0 \\ 0 & I_{22} & 0 \\ 0 & 0 & I_{33} \end{bmatrix}^B \begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix}^B +$$

$$+\begin{bmatrix} 0 & -r & q \\ r & 0 & -p \\ -q & p & 0 \end{bmatrix}^B \begin{bmatrix} I_{11} & 0 & 0 \\ 0 & I_{22} & 0 \\ 0 & 0 & I_{33} \end{bmatrix}^B \begin{bmatrix} p \\ q \\ r \end{bmatrix}^B = \begin{bmatrix} m_{B_1} \\ m_{B_2} \\ m_{B_3} \end{bmatrix}^B$$

The integration of these equations yields the body rates p, q, r .

A second set of differential equations provide the body attitudes. Three possibilities exist: Euler angle, quaternion, or direction cosine equations.

We pursue here the direction cosine approach. From the definition of the rotational derivative, Eq(4), and the Euler transformation, Eq(5), we derive directly

$$\left[\frac{dT}{dt}\right]^{BA} = [\overline{\Omega^{BA}}]^B[T]^{BA} .$$

The coordinate system A represents the local level geographic system L and the frame A is replaced by the earth frame E. The frame B is interpreted as the body frame and $]^B$ the body principle axes system:

$$\left[\frac{dT}{dt}\right]^{BL} = [\overline{\Omega^{BE}}]^B[T]^{BL} \qquad (13).$$

Expressed in coordinates:

$$\begin{bmatrix} \dot{t}_{11} & \dot{t}_{12} & \dot{t}_{13} \\ \dot{t}_{21} & \dot{t}_{22} & \dot{t}_{23} \\ \dot{t}_{31} & \dot{t}_{32} & \dot{t}_{33} \end{bmatrix} = \begin{bmatrix} 0 & r & -q \\ -r & 0 & p \\ q & -p & 0 \end{bmatrix} \begin{bmatrix} t_{11} & t_{12} & t_{13} \\ t_{21} & t_{22} & t_{23} \\ t_{31} & t_{32} & t_{33} \end{bmatrix}.$$

The Euler angles ψ, θ, ϕ are obtained from the direction cosine matrix elements

$$\theta = \arcsin(-t_{13})$$

$$\psi = \arccos\left(\frac{t_{11}}{\cos\theta}\right) sgn(t_{12})$$

$$\phi = \arccos\left(\frac{t_{33}}{\cos\theta}\right) sgn(t_{23})$$

In summary, the three translational degrees of freedom are governed by the six first order differential equations (9) and (10). The three rotational degrees of freedom are calculated from the three first order differential equations (12) and the nine kinematic differential equations (13).

Rotating Earth Model

For transatmospheric vehicles with flight speeds in the hypersonic region the Coriolis and centrifugal apparent forces cannot be neglected. The flat earth model is therefore inadequate and must be replaced by a more general model that takes into account the rotation and curvature of the earth.

We first need to relate the inertial velocity of the vehicle c.m., $[v_B^I]$, to the geographic velocity, $[v_B^E]$ for two reasons: (i) the atmosphere is rotating with the earth and the aerodynamic forces are a function of the geographic velocity, (ii) the vehicle's position and velocity are traced over the earth.

The position of the inertial reference frame *I* is oriented in the solar ecliptic and one point, *I* is co-located with the center of the earth. The earth frame *E* is fixed with the geoid and rotates with the angular velocity $[\omega^{EI}]$. By definition the inertial velocity is $[v_B^I]=[D^I s_{BI}]$, where $[s_{BI}]$ is the location of the vehicle c.m. wrt the center of the earth. To introduce the geographic velocity we change the reference frame

$$[D^I s_{BI}]=[D^E s_{BI}]+[\Omega^{EI}][s_{BI}]$$

and introduce a reference point on the earth (any) *E*

$$[s_{BI}]=[s_{BE}]+[s_{EI}]$$

$$\therefore \quad [D^E s_{BI}]=[D^E s_{BE}]+[D^E s_{EI}]=[D^E s_{BE}]=[v_B^E]$$

$[D^E s_{EI}]$ is zero because $[s_{EI}]$ is constant in the earth frame. Substituting we obtain a relationship between the inertial and geographic velocities

$$[v_B^I]=[v_B^E]+[\Omega^{EI}][s_{BI}] \tag{14}$$

Shortly, we will use Newton's law to calculate the geographic velocity in body axes $[v_B^E]^B$. Then we use Eq(14) to calculate the inertial coordinates of the vehicle

$$[ds_{BI}/dt]^I=[\overline{T}]^{BI}[v_B^E]^B+[\Omega^{EI}]^I[s_{BI}]^I$$

which requires the coordinate transformation matrix $[T]^{BI}$ of the body axes wrt the inertial axes that will be obtained from the direction cosine equations.

The translational differential equations are derived from Netwon's law, Eq(8). It must be modified to calculate the geographic velocity. We first deal with the rotational derivative and transform it to the *E* frame

$$[D^I v_B^I]=[D^E v_B^I]+[\Omega^{EI}][v_B^I] \tag{15}$$

Then we substitute Eq(14) into the transformed rotational derivative

$$[D^E v_B^I]=[D^E v_B^E]+\left[D^E([\Omega^{EI}][s_{BI}])\right]$$

where the last term is subject to the chain rule

$$\left[D^E([\Omega^{EI}][s_{BI}])\right]=[D^E\Omega^{EI}][s_{BI}]+[\Omega^{EI}][D^E s_{BI}]$$

The first term on the right hand side is zero because the earth's angular velocity is constant. The second term can be expanded

$$\begin{aligned}[\Omega^{EI}][D^E s_{BI}] &= [\Omega^{EI}]([D^E s_{BE}]+[D^E s_{EI}]) \\ &= [\Omega^{EI}][D^E s_{BE}]=[\Omega^{EI}][v_B^E]\end{aligned}$$

Collecting terms for $[D^E v_B^E]$ and substituting Eq(14) into Eq(15) we get

$$[D^I v_B^I]=[D^E v_B^E]+2[\Omega^{EI}][v_B^E]+[\Omega^{EI}][\Omega^{EI}][s_{BI}]$$

Furthermore, taking the perspective of the geographic velocity from the vehicle's body frame, we make an Euler transformation

$$[D^E v_B^E]=[D^B v_B^E]+[\Omega^{BE}][v_B^E].$$

Now we are ready to replace the rotational derivative in Newton's law, Eq(8), and obtain the translational equations of motion

$$\begin{aligned}[D^B v_B^E]+[\Omega^{BE}][v_B^E] &= \frac{1}{m}[f_{a,p}]+[g]- \\ &-2[\Omega^{EI}][v_B^E]-[\Omega^{EI}][\Omega^{EI}][s_{BI}]\end{aligned} . \tag{16}$$

Comparing this equation with the flat earth model, Eq(9), we identify the two additional terms as Coriolis and centrifugal accelerations.

The equations are valid in any coordinate system. For coding it is best to express forces, rates and velocity in body axes, but the earth's rotation, gravity and vehicle position in inertial axes

$$\begin{aligned}&[dv_B^E/dt]^B+[\Omega^{BE}]^B[v_B^E]^B=\frac{1}{m}[f_{a,p}]^B+[T]^{BI}[g]^B- \\ &-2[T]^{BI}[\Omega^{EI}]^I[\overline{T}]^{BI}[v_B^E]^B-[T]^{BI}([\Omega^{EI}]^I[\Omega^{EI}]^I[s_{BI}]^I)\end{aligned}$$

where

$$[\Omega^{BE}]=[\Omega^{BI}]-[\Omega^{EI}].$$

$[\Omega^{BI}]$ is obtained from Euler's equation, and $[T]^{BI}$ from the direction cosine equation.

A second set of differential equations solves for the inertial position $[s_{BI}]$:

$$[D^I s_{BI}]=[D^E s_{BI}]+[\Omega^{EI}][s_{BI}].$$

The inertial point *I* is located at the center of the earth and coincides with the earth's center *E* which is a particular point of the rotating earth frame *E*. Therefore, they are interchangeable:

$$\begin{aligned}[D^I s_{BI}] &= [D^E s_{BE}]+[\Omega^{EI}][s_{BI}] \\ &= [v_B^E]+[\Omega^{EI}][s_{BI}]\end{aligned}$$

and we arrived at the second set of translational equations

$$[D^I s_{BI}]=[v_B^E]+[\Omega^{EI}][s_{BI}]. \tag{17}$$

In coordinates:

$$[ds_{BI}/dt]^I=[\overline{T}]^{BI}[v_B^E]^B+[\Omega^{EI}]^I[s_{BI}]^I.$$

The two sets of component equations may also be found in Stevens and Lewis[7], Equation (1.3-6) and Equation (1.3-2), expressed in vectör notation.

The derivation of the rotational equations for a rotating earth holds no surprise. We can follow the flat earth approach with some modifications. From Eq(11), with Euler's law referred to the inertial frame *I* we get

$$[D^I l_B^{BI}]=[m_B]$$

where $[l_B^{B\,I}] = [I_B^B][\omega^{BI}]$. Transferring the rotational derivative to the body frame,

$$[D^B l_B^{B\,I}] + [\Omega^{BI}][l_B^{B\,I}] = [m_B]$$

and substituting for the angular momentum results in the rotational differential equations

$$[I_B^B][D^B\omega^{BI}] + [\Omega^{BI}][I_B^B][\omega^{BI}] = [m_B]. \qquad (18)$$

Remember that

$$[\omega^{BI}] = [\omega^{BE}] + [\omega^{EI}].$$

For coding we express the equations in body coordinates

$$[I_B^B]^B[d\omega^{BI}/dt]^B + [\Omega^{BI}]^B[I_B^B]^B[\omega^{BI}]^B = [m_B]^B.$$

The kinematic equations can also be adopted from Eq(13)

$$\left[\frac{dT}{dt}\right]^{BI} = [\overline{\Omega^{BI}}]^B[T]^{BI} \qquad (19).$$

In summary, the three translational degrees of freedom are governed by the six first order differential equations (16) and (17). The three rotational degrees of freedom are calculated from the three first order differential equations (18) and the nine kinematic differential equations (19).

Spinning Missile Model

Spin stabilized missiles require a special set of 6 DOF equations that de-couple the fast roll rate from the pitch and yaw dynamics. A so-called non-spinning reference frame B' is introduced that slips like a shell over the spinning body. Its justification comes from the aerodynamic modeling of spinning symmetrical bodies that gives rise to such terms as the Magnus lift and moment coefficients. For brevity, we make the flat earth assumption.

The translational equations are directly portable from the flat earth equations (9) and (10) by just substituting B' for B.

$$m[D^{B'}v_B^E] + m[\Omega^{B'E}][v_B^E] = [f_{a,t}] + m[g] \qquad (20)$$

with the coordinate version expressed in non-spinning body axes

$$m[dv_B^E/dt]^{B'} + m[\Omega^{B'E}]^{B'}[v_B^E]^{B'} = [f_{a,t}]^{B'} + m[T]^{B'L}[g]^L$$

The $]^{B'}$ coordinates are obtained from the local level coordinates by only two rotations, yaw and pitch. Thus the second axis lies always in the horizontal plane.

The kinematic equations, are the unmodified Eq(10)

$$[D^E s_{BE}] = [v_B^E] \qquad (21)$$

and its modified coordinate form:

$$[ds_{BE}/dt]^L = [\bar{T}]^{B'L}[v_B^E]^{B'}.$$

The derivation of the rotational equations starts with Eq(11) transferred to the non-rotating body frame

$$[D^{B'}l_B^{B\,E}] + [\Omega^{B'E}][l_B^{B\,E}] = [m_B].$$

With the angular momentum now divided into two parts

$$[l_B^{B\,E}] = [I_B^B][\omega^{BB'}] + [I_B^B][\omega^{B'E}].$$

We particularize the rotational derivative

$$\begin{aligned}[D^{B'}l_B^{B\,E}] &= \left[D^{B'}([I_B^B][\omega^{BB'}])\right] + \left[D^{B'}([I_B^B][\omega^{B'E}])\right]\\ &= [I_B^B][D^{B'}\omega^{BB'}] + [I_B^B][D^{B'}\omega^{B'E}]\end{aligned}$$

where we made the assumption that the missile has rotational symmetry (at least on the average) and, therefore, the rotational derivative wrt the non-spinning body frame of the moment of inertia tensor is zero. Thus the rotational equations of motion are

$$\begin{aligned}&[I_B^B][D^{B'}\omega^{BB'}] + [I_B^B][D^{B'}\omega^{B'E}] +\\ &+ [\Omega^{B'E}][I_B^B][\omega^{BB'}] + [\Omega^{B'E}][I_B^B][\omega^{B'E}] = [m_B]\end{aligned} \qquad (22)$$

The first term models the change in angular momentum caused by variable spin rate of the missile. For a missile with constant spin rate, this term is zero. The second term represents the change of angular momentum of the nonrating body shell as it is subjected to yaw and pitch rate changes. The gyroscopic coupling moment is expressed by the third term which usually dominates the forth term.

Expressed in non-rotating body coordinates

$$\begin{aligned}&[I_B^B]^{B'}[d\omega^{BB'}/dt]^{B'} + [I_B^B]^{B'}[d\omega^{B'E}/dt]^{B'} +\\ &+ [\Omega^{B'E}]^{B'}[I_B^B]^{B'}([\omega^{BB'}]^{B'} + [\omega^{B'E}]^{B'}) = [m_B]^{B'}\end{aligned}$$

Carefully distinguish between B and B'. The point B is always the c.m. of the missile and the coordinate system is always the non-rotating $]^{B'}$; but the frame may be the body fixed frame, as in $[I_B^B]$, or the non-rolling frame, as in $[\omega^{B'E}]$.

The matrix form of these equations for

$$[I_B^B]^{B'} = \begin{bmatrix} I_{11} & 0 & 0 \\ 0 & I_{22} & 0 \\ 0 & 0 & I_{22} \end{bmatrix}^{B'}, [\omega^{BB'}]^{B'} = \begin{bmatrix} \omega \\ 0 \\ 0 \end{bmatrix}^{B'}, [\omega^{B'E}]^{B'} = \begin{bmatrix} p \\ q \\ r \end{bmatrix}^{B'}$$

are

$$\begin{bmatrix} I_{11}\dot{\omega} \\ 0 \\ 0 \end{bmatrix}^{B'} + \begin{bmatrix} I_{11}\dot{p} \\ I_{22}\dot{q} \\ I_{22}\dot{r} \end{bmatrix}^{B'} + \begin{bmatrix} 0 & -r & q \\ r & 0 & -p \\ -q & p & 0 \end{bmatrix}^{B'} \begin{bmatrix} I_{11} & 0 & 0 \\ 0 & I_{22} & 0 \\ 0 & 0 & I_{22} \end{bmatrix}^{B'} \times$$

$$\times \left(\begin{bmatrix} \omega \\ 0 \\ 0 \end{bmatrix}^{B'} + \begin{bmatrix} p \\ q \\ r \end{bmatrix}^{B'} \right) = \begin{bmatrix} m_{B_1} \\ m_{B_2} \\ m_{B_3} \end{bmatrix}^{B'}$$

Multiplying out these matrices provides us with the three scalar differential equations

$$I_{11}(\dot{\omega} + \dot{p}) = m_{B_1}$$

$$I_{22}\dot{q} + I_{11}\omega r - pr(I_{22} - I_{11}) = m_{B_2}$$

$$I_{22}\dot{r} - I_{11}\omega q + pq(I_{22} - I_{11}) = m_{B_3}$$

with the gyroscopic coupling terms $I_{11}\omega r$ and $I_{11}\omega q$.

The kinematic equations are obtained by replacing B with B' in Eq(13)

$$\left[\frac{dT}{dt}\right]^{B'L} = [\overline{\Omega^{B'E}}]^{B'}[T]^{B'L} \qquad (23)$$

The complete set of 6 DOF equations of motions consists of Eq(20) through (23) with their corresponding coordinate expressions that are suitable for computer programming.

Simulations

The three sets of 6DOF equations served as typical examples of the tensor modeling process that leads straight to programmable matrix equations. These equations are so well known from text books that our derivation here may seem to overstate the case for tensor modeling.

We found however that for difficult modeling tasks our systematic approach has certain advantages. We built a computer simulation for *transatmospheric vehicles* that required a spherical rotating earth model. We had to deal with the inertial frame, the atmosphere fixed reference frame (for aerodynamics and winds) and the body frame of the vehicle. Yet the equations had to be programmed in geographic coordinates.

Complicating the task were the navigation and guidance components of the vehicle. Ground radar provides vehicle and target states relative to the earth, while on-board sensors measure the target states relative to the vehicle.

Working from tensor models to matrix computer code prevented us from getting lost in too many component equations. An application of these equations is provided in Reference 8.

Simulating *air-to-air engagements* requires the modeling of at least three independent vehicles, the shooter, the target and the intercepting missile. These vehicles are related to each other by acquisition radars, data links and missile seeker trackers. We lost count of the number of frames and coordinate systems that are required for modeling such an engagement. Again, our systematic approach helped us to develop simulations that are used by several U.S. and foreign agencies for missile performance evaluations and in manned simulators[9].

Vehicle simulations that run in a real time environment, like flight simulators, must be programmed with fast executing code. Subroutine calls should be avoided if possible. FORTAN 77 has no intrinsic matrix functions and therefore subroutines have to be used to manipulate matrices. This disadvantage is offset however by the compactness of the code and the ease of debugging. Fortran 90 alleviates this shortcoming with the built-in intrinsic matrix functions.

Students at the Graduate Engineering and Research Center of the University of Florida have been offered a course in Tensor Modeling of Flight Dynamics since the late 70'. Currently, this course is taught under the title: *Modeling and Simulation of Aerospace Vehicles.* After an initial cultural shock they discover the advantages of tensor modeling over vector mechanics and begin to appreciate the difference between frames and coordinate systems, the usefulness of the rotational derivative and the Euler transformation. By the end of the semester they are proficient in the art of modeling and will have worked on computer simulations from 1 DOF through 6 DOF.

Conclusions

The two steps approach -- from tensor modeling to matrix computer code -- has served us well over two decades. The tensor formulation of flight dynamic problems highlights the key physical features unburdened by coordinate systems. Once the necessary transformations and simplifications are made, the appropriate coordinate systems are selected and the matrix equations are programmed.

Over the years simulations were developed in all categories - from spinning bodies to endo and exo atmospheric missiles - using this technique.

Students at the University of Florida are given a fresh look at flight dynamics and hands-on experience in developing flight simulations.

References

[1] American National Standard, "*Recommended Practice for Atmospheric and Space Flight Vehicle Coordinate Systems*" ANSI/AIAA R-004-1992.

[2] Zipfel, P.H. "*Modeling and Simulation of Aerospace Vehicles*", University of Florida Course Notes EAS 6939, (1996).

[3] Wrede, R.C. "Introduction to Vector and Tensor Analysis", 1963 Wiley.

[4] Zipfel, P.H., "*On Flight Dynamics of Magnus Rotors*", Nov 1970. Ph.D. Dissertation (AD 716 345).

[5] Synge, J.L. "*Classical Dynamics*", in Handbuch der Physik Band III/1, Springer, Berlin, p 9 (1960).

[6] CADAC-PC "Computer Aided Design of Aerospace Concepts for PC", Tybrin Co, March 1995

[7] Stevens, B.L., Lewis, F.L., *Aircraft Control and Simulation* John Wiley, 1992

[8] Zipfel, P.H., "*Guidance Laws for Hypersonic Vehicles*", USAF/ASD/XR Concept Study #100, March 1991.

[9] Zipfel, P.H.,"*Seeker and Guidance Modeling for the CADAC4 Air-to-Air Trajectory Program*' USAF, AD/XR Concept Study #48, June 1983

MODIFIED VISTA/F-16 PILOTED SIMULATION STUDY

Phillip D. McKeehen*, Thomas J. Cord', and Ba T. Nguyen'
Wright Laboratory (WL/FIGC)
Wright Patterson Air Force Base, Ohio 45433-7531

ABSTRACT

A piloted experimental study of potential enhanced task performance resulting from improved high-angle-of-attack aerodynamic and flight control capability was conducted in the Wright Laboratory's engineering flight simulator facility. The simulation database used was representative of the aerodynamics and inertias of the Variable-stability In-flight Simulator Test Aircraft (VISTA) / F-16. The VISTA variable-stability system was not used. This distinction should be noted whenever the VISTA is referred to in this paper. Flight test pilots evaluated both baseline and three modified versions of the simulated aircraft using a variety of high-angle-of-attack tasks. Aerodynamic modifications were based on wind tunnel data from a previous effort which examined various means of extending the aircraft angle-of-attack limits. These focused primarily on the lateral-directional characteristics in the twenty-nine to thirty-seven degree range. Flight control modifications came from a new approach to control of lateral-directional dynamics which used variable structure control and describing functions. This controlled the forebody vortices to achieve improved roll coordination. The various configurations were compared relative to their departure resistance and maneuverability. This paper presents some quantitative and qualitative results of the study which show improved departure resistance and mixed results with respect to maneuverability.

INTRODUCTION

The Control Dynamics Branch of the USAF Wright Laboratory has been involved in high-angle-of-attack (AOA) flight research for the last several years from the perspectives of aircraft dynamics, aerodynamics and flight control. The effort reported on here was a piloted simulation study of a configuration that represents the culmination of two previous efforts involving aerodynamic and related flight control law modifications to high-performance fighter designs.

Aerodynamic modifications In [2], Simon et al performed a wind-tunnel test program to define the aerodynamics of an F-16 with (i) cut-back wing leading-edge extension (LEX), (ii) forebody chines and (iii) pneumatic forebody vortex control as shown in Figure 1.

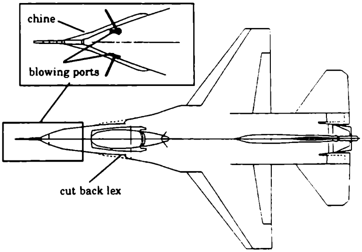


Figure 1 Active and Passive Modifications

The high-AOA effects sought were, i) improved directional stability from the use of chines, ii) improved longitudinal stability and elimination of the high-AOA pitch trim point by a cut-back LEX, and iii) increased directional control power by means of forebody

[*] Senior Member, AIAA
['] Flying Qualities Engineer

vortex control. Figures 2 through 6 show the specific beneficial changes to the total pitching, rolling and yawing moments due to the passive (i.e., no active vortex control) modifications as well as the rolling and yawing moments resulting from the active vortex control.

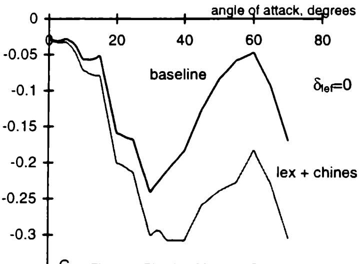


Figure 2 Pitching Moment Comparison
Mach .3, alt10K, c.g. 40%

The well-known F-16 tendency towards hung stall in the vicinity of sixty-degrees AOA is also eliminated by the effective nose-down pitching moment increment produced by reduction of the LEX area forward of the center of gravity. This increment is shown in Figure 2.

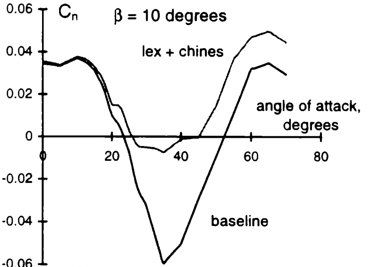


Figure 3 Yawing Moment Comparison,
Mach .3, alt 10K, c.g. 40%

The increase in yawing moment above twenty-degrees AOA generated by the chines, see Figure 3, reduces the static directional instability.

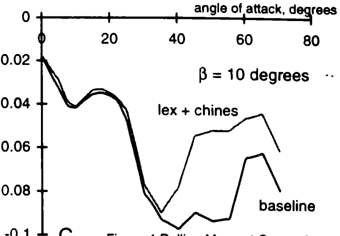


Figure 4 Rolling Moment Comparison
Mach .3, Alt 10K, c.g. 40%

The effectiveness of vortex control in generating yawing moment is shown in Figure 5. The associated incremental rolling moment, see Figure 6, is predictable and proverse through approximately thirty-five degrees and is noted in [3] as the limiting factor in the control of sideslip.

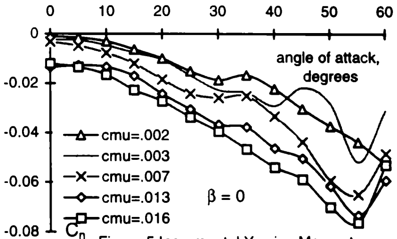


Figure 5 Incremental Yawing Moment

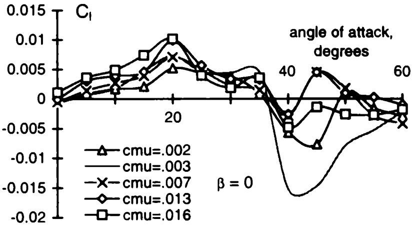


Figure 6 Incremental Rolling Moment

Control law implementation Reference 3 describes the development of control laws for a forebody vortex control system which would augment yaw control power as the rudder loses effectiveness at high AOA. Figure 7 shows the control law block diagram from that report. Adams and Buffington placed the control of the forebody vortex as an outside loop around the existing F-16 Block 40 control laws. This control was implemented as a bang-bang system with three possible states: full blowing from the left forebody slot, full blowing from the right forebody slot or no blowing. The bang-bang system was chosen because it was considered to be the most conservative. σ(x) represents a switching surface that is a function of aircraft states and has δ as a deadband. Blowing from the left occurs when σ(x) is less than negative δ, right blowing is for σ(x) greater than positive δ and no blowing will happen as long as σ(x) is within the deadband. Design of the system was accomplished using variable structure control and describing functions and is described in detail in [3]. Reference 3 also gives a detailed summary of the non-realtime simulation study of potential high-AOA flight stability and performance enhancements. For this effort, high AOA is defined as flight up to the maximum-lift AOA or approximately 35 degrees. Just as the wind tunnel tests of [2] validated the predicted improvements in directional stability and directional control power, the results in [3] showed that use of the modifications allow the extension of the VISTA/F-16's AOA envelope to 37 degrees AOA. The same non-realtime simulation used in [3] was implemented in WL/FIGD's piloted simulation facility in the summer of 1995 and a more realistic piloted simulation study was then conducted in mid-September 1995.

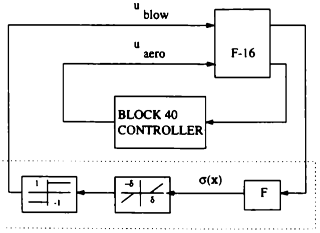


Figure 7 Forebody Blowing Control System

AIRCRAFT CONFIGURATIONS

Based on the modifications to the baseline VISTA's aerodynamics defined above, four distinct configurations were flown during the study:

Baseline = baseline VISTA with 29 degree AOA limiter

Extended = baseline VISTA with limiter extended to 40 degrees

LEX/Chines = baseline VISTA with limiter extended to 40 and cut-back LEX and chines

LEX/Chines/Blowing = baseline VISTA with limiter extended to 40 and cut-back LEX and chines with active vortex blowing controller

These configurations are shown as BASE, EXT, LC and LCB, respectively, in the following discussion. EXT, LC, and LCB are the same configurations used in reference 3.

EXPERIMENT SET-UP

Simulation techniques for investigating departure characteristics are fairly straightforward, involving either deliberate excursions into portions of the flight envelope in which the aircraft being investigated is suspected of being departure prone or defining pilot tasks which may result in unstable dynamic conditions. The approach to quantifying the various aspects of maneuverability is a little more complex in that the balance between aerodynamic performance and operational effectiveness depends on where you are in the aircraft's life cycle. Since this was to be a reasonably small effort, elements of aerodynamics, flying qualities and operationally-relevant tasks were chosen to demonstrate the maneuverability levels. Performance tasks were basically open-loop with the goal of quantifying roll capability and sideslip response at various angles of attack. Flying qualities was limited to the conduct of two simple tasks, a split-S and a loaded roll reversal. The operational relevancy was demonstrated by target acquisition at high angle of attack. The data gathered here stops well short of allowing prediction of exchange ratios in m-on-n air combat, but does satisfy the goal of saying on the most basic level if the aircraft-pilot system has increased potential to succeed in a fight.

Simulator capabilities The piloted portion of this study was conducted at the United States Air Force Wright Laboratory using the Large Amplitude Multimode Aerospace Research Simulator (LAMARS). The LAMARS is a five-degree-of-freedom, beam-type, motion-base simulator used for flight control system development and flying qualities research. For this study, the motion capability of the LAMARS was not utilized.

The aircraft cockpit is housed within a 20 foot diameter sphere at the end of the beam. The cockpit instrumentation and controls are based on an F-16D configuration. A Compuscene IVa computer system along with three liquid crystal display projectors provided a 120 degree by 40 degree vertical visual scene for the pilot. Also, a raster projection system was used to display F-16 heads-up display (HUD) information onto the surface of the dome. Finally, for the target acquisition portion of the study, a laser target projector was used to display a calligraphic image of the target that was to be acquired by the pilot.

The aerodynamics and flight control modifications were implemented into the real-time simulation framework. The real-time model is a high-fidelity, six-degree-of-freedom aircraft mode based on Lockheed Martin VISTA/F-16 data. It is composed of several real-time components: a VISTA/F-16 aerodynamic model, an F110-GE-100 engine model, a Multi-Axis Thrust Vectoring (MATV) flight control model, and a third-order actuator model of the Integrated Servo Actuators used in the VISTA/F-16. The thrust vectoring portion of the flight control system as well as the variable stability system were disabled for this study. The resulting flight control system utilized the Block 40 F-16 flight control laws for the baseline configuration. All components of the model were run at 50 hertz with the exception of the 100 hertz actuator model.

Pilots The pilots used for the experiment were highly-qualified veteran F-16 flight test pilots assigned to the Air Force Flight Test Center at Edwards AFB and Wright-Patterson's F-22 SPO. This was considered important both for their ability to perform the experiment as designed and for their ability to comment on the results with respect to application of the enhanced capability to operational use.

Tasks The tasks required of the pilots during the study were divided into performance, flying qualities and target acquisition evaluation tasks. Tasks 1 through 3 are the same as those used in reference 3. Tasks 4 and 5 are modified STEMs from reference 5. The tasks are described below.

Performance:

Task1 (Max Lateral Stick Response): From a trimmed condition, the pilot aggressively pitched the aircraft up to the desired AOA and applied a two-second full right lateral stick pulse while holding the test AOA.

Flying qualities evaluation:

Task2 (Split-S): From a trimmed condition, the pilot pitched the aircraft up to the desired AOA. He then rolled 180 degrees while maintaining AOA. He continued to pull through to achieve a 180-degree heading change.

Task3 (Loaded Roll Reversal): The pilot entered a 2-g right turn. He then pitched up to the desired test AOA He then applied a left lateral stick pulse to capture approximately -60 degree bank angle while holding the test AOA.

Target Acquisition:

Target set-up: One of the pilots flew the baseline configuration from 1-g level trim into a constant AOA 3-g descending turn. The results were recorded and used as the target aircraft for tasks 4 and 5.

Task4 (Gross Lateral Acquisition): The test aircraft began in 1-g level flight approximately 1500 ft behind and 1000 feet below the target aircraft. When the target rolled, the pilot hesitated until the target was approximately 10 to 20 degrees off of the nose. He then quickly pulled to the test AOA, hesitated momentarily, then rolled aggressively while holding AOA to capture the target.

Task5 (Gross Longitudinal Acquisition) : The test aircraft began in 1-g level flight approximately 3000 ft directly behind the target aircraft. The pilot allowed the target to reach a predetermined angle off the nose, then rolled to get into the target's maneuver plane. He then hesitated until he was in a lag position such that the test AOA occurred at target capture.

In addition, free-form maneuvering was conducted against a recorded target performing both constant-g, level turns and non-cooperative, three-dimensional trajectories. These conditions were chosen to investigate the effect of improved roll capability on Close-In Combat. This data is still being analyzed and will be reported on at a later date.

Flight conditions The tasks mentioned above were performed for a range of trim conditions from reference 3 with Mach numbers between .3 and .7 and altitudes between 10,000 and 25,000 ft. The test AOAs (AOAt) ranged from 20 degrees to 35 degrees.

PRELIMINARY RESULTS

The two-week study produced a large volume of data to be analyzed. The results shown here represent some aggregate statistics from approximately one-third of the simulation runs performed.

Departure resistance The primary objective of both the non-realtime simulation study done in [3] and this study was to determine the feasibility of expanding the VISTA/F-16's AOA envelope and thus increase the range of AOAs that it might simulate. The key to this is to enhance the vehicle's departure resistance up through the unstable 30 - 35 degree AOA region. Based on the data analyzed to date, the following table summarizes the relative departure resistance of the various configurations.

Configuration	Departures	Flights
BASE	0	3
EXT	10	25
LC	0	25
LCB	0	25

Table1 Departure Occurrences

As is clearly evident, the modifications do allow the VISTA to fly to maximum lift AOA without departure. The following are typical pilot comments underscoring the benefits of the LC and LCB configurations.

Task 2 m=.5. alt=10k ft. test AOA=30 deg

EXT: "Directional stability went away. Alpha way too high."
LC: "Not quite as good as [LCB]. Not quite as fast rolling and pitch rate slower."
LCB: "I liked that a lot."

Task3 m=.4,alt=25k ft, test AOA=35 deg

EXT: "Off to the races! Coupled lateral input-pitch departure."
LC: "Trouble maintaining alpha. Roll performance better than [extended]."
LCB: "Got there. Slow, but got there.. Controllable."

Maneuverability In addition to departure resistance, this study addressed the issue of whether the modifications would yield greater maneuverability of the aircraft. Based on the tasks performed here, two measures of merit for the various configurations' maneuverability were defined: (i) capture time and (ii) proposed maneuverability (PM) metric. The PM metric, elements of which are shown in Figure 8, is defined as the capture time multiplied by (total) bank angle error, i.e. the widest peak to trough excursion of the bank angle during the rolling part of this task. This metric is proposed because the capture time did not correlate with expectations or the pilots' commentary. It is an attempt to factor in both speed of rolling and accuracy with respect to the size of overshoots in bank angle. Note that both capture time and the PM metric are task dependent, as follows:

Task2: Capture time is defined for the Split-S maneuver as the time from roll initiation to pull through a nose-low attitude.

Task3: Capture time is defined for the loaded roll reversal as time from start of roll reversal until a steady-state negative bank angle has been achieved within a plus or minus ten degree tolerance.

Task4: Here the pilot does actually laterally acquire a target aircraft. Capture time is defined as the time from pull up initiation to target capture. Target capture was not recorded às a specific event, but is approximated by reaching a steady state bank angle within a twenty-degree total tolerance.

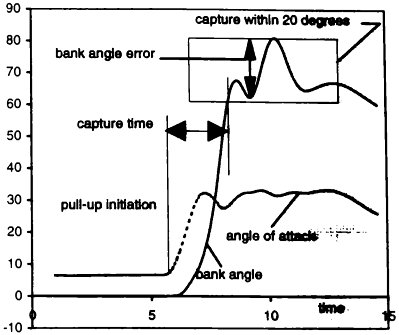


Figure 8 PM elements for Task 4

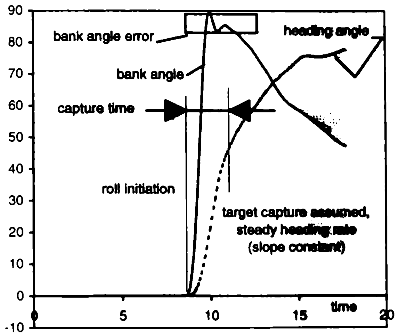


Figure 9 PM elements for Task 5

Task5: Capture time for the gross longitudinal acquisition is again approximated, here as time from roll initiation to steady state heading angle rate. The PM metric is defined as in task 4 except that the roll error is measured at the beginning of the maneuver rather than at the capture, see Figure 9.

Improved stability and control characteristics were noted in the wind tunnel tests in [2]. Improved departure resistance up to maximum lift AOA was noted in the non-realtime simulation studies performed in [3]. These same trends have been seen in the current study for the active and passive modifications to the VISTA aircraft. The expected maneuverability trend was that capture times would decrease as the configuration was changed from BASE to EXT to LC to LCB. In actuality, as shown in the following plots, the expected trend did not always occur.

Figure 10 shows a typical result obtained for the Split-S maneuver, where the straight lines are superimposed for trend clarity. Based on wind tunnel tests it may be predicted that decreased sideslip associated with the more stable LC and LCB configurations would result in faster roll rates and thus lower capture times than the EXT configuration. Indeed, the EXT configuration departed for all three pilots. Slightly lower capture times were obtained for the LC configuration than for the LCB.

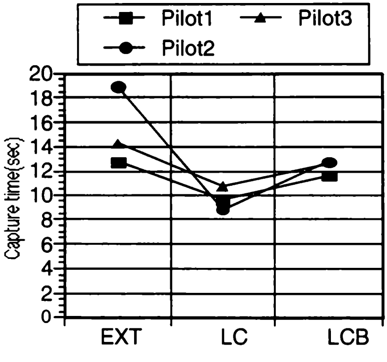


Fig 10 Task 2 M=.5 AOAt=35deg

Figure 11 shows less pilot consistency with respect to the different configurations than does Figure 10. For the Task 3 results, pilot 2 was the only pilot whose results were in accordance with the corresponding theoretical predictions, namely that the LC configuration would provide lower capture times than the EXT and the LCB would provide even lower capture times than the LC did. Pilot 1 shows the reverse trend from prediction, although he departed shortly after completing the task, using the EXT configuration. Pilot 3 shows better capture time for EXT than for either LC or LCB (which for this pilot is better than LC).

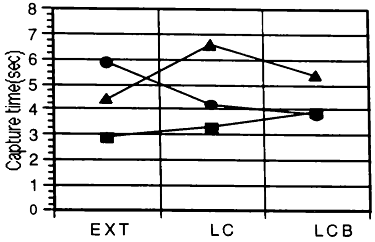


Fig 11 Task 3 M=.4 AOAt=35deg

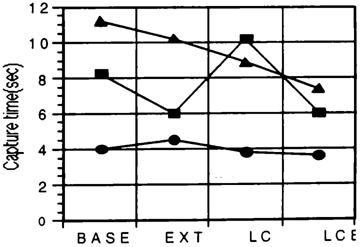


Fig 12 Task 4 M=.5 AOAt=25deg

Figure 12 shows that pilot 2 noticed very little difference with respect to configurations. Pilot 1 improves with EXT over BASE then shows degraded performance with LC and improvement again with LCB. Pilot 3 shows the capture time behavior to be expected-consistent decrease from BASE to EXT to LC to LCB.

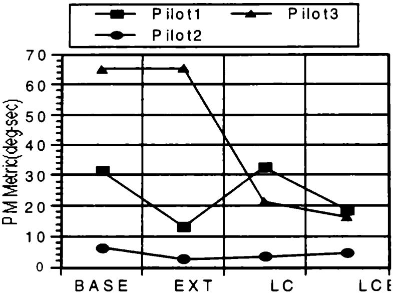


Fig 13 Task 4 M=.5 AOAt=25deg

Figure 13 then gives the PM metric to also account for accuracy for the same task and flight conditions. Once the accuracy factor is also added to the analysis, pilot 1 and 2 results show no significant difference in trend from the capture time plots. However, pilot 3 shows marked improvement for the LC and LCB configurations over the BASE and EXT.

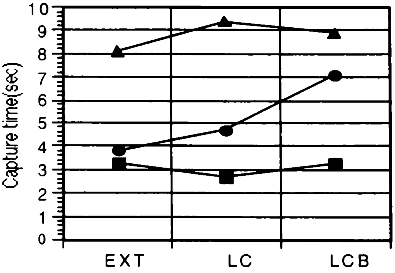


Fig 14 Task 4 M=.5 AOAt=35deg

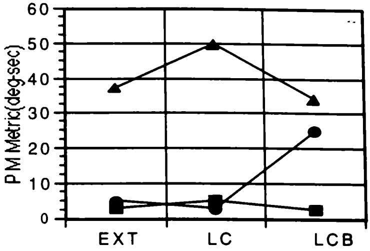


Fig 15 Task 4 M=.5 AOAt=35deg

Figures 14 and 15 show similar trends for Task 4 when the test AOA is raised to 35 degrees. In both these plots, little difference is shown by pilot 1 in flying the 3 configurations. For pilot 3 results are a little worse for the LC than EXT or LCB. Pilot 2 shows a significant rise in capture time and the PM metric once the blowing is part of the configuration.

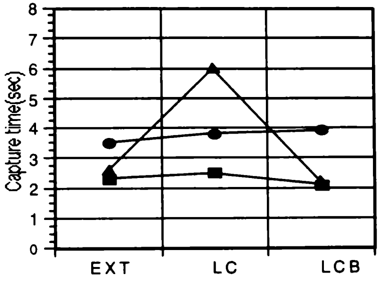


Fig 16 Task 5 M=.5 AOAt=35deg

A similar pattern of results is evident in Figures 16 and 17. Again pilot 1 shows little difference in results for the 3 configurations. Pilot 2 shows only a very slight degradation when progressing from EXT to LC to LCB. Pilot 3 again performs better with both EXT and LCB than with the LC configuration.

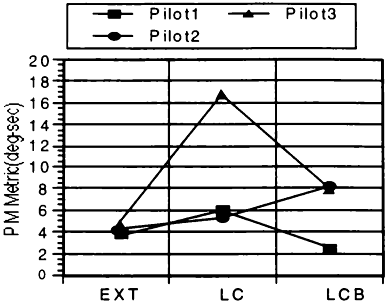


Fig 17 Task 5 M=.5 AOAt=35deg

CONCLUSIONS

This experimental simulation study was designed to determine, both quantitatively and qualitatively, the potential benefits to be gained in the VISTA/F-16's high-AOA performance due to the use of some simple aerodynamic modifications. The modifications included a cut-back wing LEX, nose chines and pneumatic forebody vortex control. The study included both an assessment of departure resistance and maneuverability. Based on the data analyzed so far from the study, the corresponding conclusions are:

(1) Departure resistance is definitely increased up to the maximum lift AOA by using the modifications. Wind tunnel studies predict that the LC configuration should be an improvement over the EXT and the LCB an improvement over the LC in high-AOA stability and departure resistance. This has been shown for both the non-realtime simulation research in [3] and this study.

(2) It is difficult to quantify maneuverability improvements (using the capture time and the PM metric defined in this study) resulting from the aerodynamic and flight control modifications. Possible reasons for this are: (a) normal pilot variability, (b) learning curve associated with the higher-AOA flight regime, and (c) the use of a non-optimum flight control system for maneuvers selected. Qualitatively, the LC and LCB configurations were described as more capable.

(3) In a more global sense, it is clear that the increased angle-of-attack capability gives the pilot more options to solve the tactical problems encountered in air-to-air combat. It is still unclear as to how to describe what the advantage is and how to say how much of it is needed. Better metrics are required as is more testing which ties those metrics to operational advantage.

ACKNOWLEDGMENTS

The authors wish to express their appreciation to Wright Laboratory Control Integration and Assessment Branch engineers Brian Stadler and Joe Nalepka for their work in implementing the aerodynamic and flight control system modifications into the real-time F-16 simulation framework as well as for their help in planning and operating the LAMARS during piloted testing.

REFERENCES

[1] McKeehen, P. D., "F-16/VISTA High-Angle-of-Attack Aerodynamic Modeling and Analysis", WL-TM-91-304, Wright Laboratory, Wright-Patterson AFB OH, February 1991.

[2] Simon, J. M., LeMay, S., Brandon, J. M., "Results of Exploratory Wind Tunnel Tests of F-16/VISTA Forebody Vortex Control Devices", WL-TR-93-3013, Wright Laboratory, Wright-Patterson AFB OH, January 1993.

[3] Adams.R. J., Buffington, J. M., "Design and Analysis of Modifications for VISTA F-16 High Angle-of-Attack Envelope Expansion", WL-TR-93-3064, Wright Laboratory, Wright-Patterson AFB OH, July 1993.

[4] McKeehen, P.D. ,"GENESIS Simulation of a Modified VISTA/F-16",AIAA-95-3381-CP, presented at AIAA Flight Simulation Technologies Conference, August 1995.

[5]Wilson, D.J., Riley, D.R., and Citurs, K.D., "Aircraft Maneuvers for the Evaluation of Flying Qualities and Agility - Maneuver Descriptions and Selection Guide," WL-TR-93-3082, August 1993.

AIAA-96-3509-CP

Experiment Control Station for the SIMONA Research Simulator

S.K. Advani [1], B. van der Geest [2], A. Wattimury [2], P.C.A. van Gool[3]
International Centre for Research in
Simulation, Motion and Navigation Technologies SIMONA
Kluyverweg, 2629 HS Delft, The Netherlands
tel +31 15 278 1395 fax +31 15 278 6480 E-Mail: s.advani@ lr.tudelft.nl

Abstract

This paper will outline the criteria necessary in designing an effective Experiment Control Station in a research simulator environment. While a training simulator requires an efficient user interface through the Instructor Operator Station, a research simulator also demands considerable interaction and monitoring on the part of the experiment controller. These requirements, with application to an on-board ECS, are outlined. The requirements for the experimenter's working environment are also discussed. A design example, the SIMONA Research Simulator ECS, is shown, describing the software and hardware environments. Potential applications of this simulator will include investigations into the presentation of information to the ECS (or IOS) operator. The paper concludes with a view of potential long-term developments in IOS, including the utilization of remote off-board instructors operating through an interactive network. This research is proposed under the framework of the International Centre for Research in Simulation, Motion and Navigation Technologies SIMONA, which will utilize its research simulator to conduct such investigations.

Nomenclature

ECS	Experiment Control Station
SRS	SIMONA Research Simulator
IOS	Instructor Operator Station
LOFT	Line-Oriented Flight Training

Introduction

The user interface of a complex flight simulation system is a subject of much attention. The effectiveness of the simulation operations, and the value of the simulation trial depends considerably on the human operator to interact with the simulation environment. The instructor is immersed on a considerably different level than the simulator pilot(s), and the information exchange which takes place has traditionally depended on the abilities of the instructor (and level of skill with the particular environment). Modernization in many training simulators has led to the application of lesson plans, allowing deterministic simulation trials, where the aircraft parameters are recorded for later evaluation.

Training simulators provide the student pilot with conditions which are controlled or monitored from a Instructor Operator Station. The requirements for research simulator experiment controllers to interface effectively with the simulation environment can pose even more critical requirements on the design of the Experiment Control Station (ECS). Research simulators require considerable flexibility in their software and hardware, making the job of experiment a complex task. The emphasis here is on allowing the operator access to the functions of the simulator hardware and software, to oversee the activities of the (experimental) subjects, and to monitor safety at all times.

[1] Director, SIMONA / Assistant Professor / Member AIAA
[2] Research Assistant
[3] Ph.D. Candidate / Member AIAA

The Delft University of Technology is developing the International Centre for Research in Simulation, Motion and Navigation Technologies, SIMONA[1]. This program is aimed at developing industrially-applicable simulation technologies, as well as aiding aviation authorities in establishing or revising simulation standards. Later, the simulator will be used to investigate the man-machine interactions of flight vehicles. Advanced radio navigation models, flight control laws and displays concepts will be studied in SIMONA.

The central element of this facility is the SIMONA Research Simulator (SRS), which will serve as a multi-functional platform to carry out these studies. The SRS flight-deck is integrated within a composite-materials shell which serves also as the primary load-bearing structure. Forces generated by the payloads are transferred through this structure (called "shuttle") and to the motion system. Figure 1 shows the shuttle. The application of advanced composite materials has led to a design with high acceleration capability. The motion platform mass does not exceed 3500 kg, and has a high stiffness such that the structural natural frequency exceeds 15 Hz[2]. These requirements have been motivated by the need to present the experimental pilots with motion cues, primarily acceleration-related, with bandwidths representative of those encountered in a wide range of flight vehicles. Motion bandwidth and temporal fidelity are crucial, since the device should aid in setting new standards for application in training systems. Research has shown the impact which the mass and vertical location of the mass of the moving payload, have on the dynamic response of the simulator [3, 4]. For these reasons, SIMONA selected to minimize the volume of the shuttle, and place as much of the structure beneath the plane of the motion system's upper gimbals. As a result, the internal space would be limited to that which is absolutely necessary, and the design of all on-board work spaces, including the Experiment Control Station, become an important issue.

General Requirements

The simulator described in this paper is intended for research into (a) simulation techniques, and (b) man-machine interactions in vehicle guidance and control. Note that both flight and surface vehicles will be represented. Flexibility in the hardware and software environment is therefore crucial, however the availability of the system to its wide range of users should not lead to compromises in operational safety. Operation of the simulator should be monitored at all times, and information related to the state of the systems. Some experiments demand that an experimenter be in close proximity to the subjects. Training these subjects also requires close interaction with experienced facility operators. For these reasons, it was decided to implement an on-board ECS (Slave ECS) in addition to an off-board station (Master ECS). Furthermore, research into simulation technologies, in support of simulator industrial applications, requires that the device have some compatibility with training environments. Research results from studies into instructor station effectiveness could easily be translated to improvements in industrial products by allowing an on-board experimenter to act as the experimental *subject*. Most importantly, however, in a university environment, it was felt that safety could be enhanced by monitoring the activities from an on-board station, occupied by an experienced operator.

While no experiment can be considered "typical", interactions with the entire software environment, simulator and simulated-vehicle variables, and observation of the pilots will always be required. Since simulation sessions may exceed one hour, comfort is necessary so that the experimenter's fatigue does not become a success (or safety) determining factor.

Experimental Procedures

Research projects in SIMONA will eventually lead to the carrying out of experiments in the SRS. Prior to this stage, a number of steps will have been taken by the experimenter in preparing the routines of the trials, familiarization of all personnel involved, and in ensuring that the use of the simulator will be optimum. Preparation of the experiments will include:

- Off-line simulation (non-real-time or real-time using a limited simulation capability)
- Design of ECS interface displays, if applicable
- Selection, presentation and logging of variables available in the software
- Familiarization with safety and operational procedures
- Dry-runs with SRS partially operating
- Pre-flight trials
- Briefing of subjects
- *Experimental trials*
- Data logging for post-processing
- De-briefing of subjects

Software Environment

SIMONA is developing a powerful real-time software environment which will allow its activities to be conducted with deterministic qualities, and still allow a high level of flexibility[5]. The ECS will interact directly with the software environment in all phases of operation.

System modes

The total simulation process, from login to shutdown, is characterized by states and transitions between states or modes. For examples, the transition from non-real-time to real-time operation, or enabling the motion system, and moving the shuttle from a low to high position, to actual simulation. These transitions create different simulation phases, which are reflected in the SRS software system. The software system is therefore based on nine system modes, as shown in Figure 2.

The mode transitions can be influenced by the experimenter. After login, the system will enter the initialization mode (INIT). After successively passing the configure, start and hold modes (indicated by CONFIG, START and HOLD respectively), the experimenter lets real simulation take place in the run-mode (RUN). To close the session, the experimenter consecutively will select HOLD, START and INIT or the shutdown mode, called SHUTDOWN. Note that since not every simulator function will be available in every system mode, the experimenter must always be fully aware of the current mode.

Research Versus Training Simulator Environments

It is valuable to indicate the primary differences between the environments required for training, as opposed to research simulation. An analysis of both environments have supported the design requirements for the SRS Experiment Control Station.

Generally, training simulators offer instructor stations positioned to one side of the simulator cab, although there have been recently-introduced changes to this philosophy. The instructor usually configures the simulator to follow a lesson plan which the pilots then follow. Much of the instructor functions are automated, allowing deterministic and interactions with the simulator session. Instructor activities, measured during LOFT training sessions are shown in Table 1. Note that in this case a lesson plan was not used. At least 56 percent of the instructor's time is spent observing activities in the flight-deck, while up to 34 percent of the time may be consumed while interacting with the IOS. Note that not all airlines follow the philosophy of lesson plans, thus requiring significant interaction on the part of the instructor. In addition to the functional requirements of the software enabling the instructor to interact, the ergonomic aspects are also of concern.

In summary, the major observations of the instruction and experiment control environments are:

- Experiment sessions will usually differ considerably, particularly when a new project begins. Note that several projects may take place during the course of one day.
- With the main task of instructors being to train pilots, these instructors are familiar with the expected actions of the pilots. Instruction may involve varying levels of interjection on the part of the trainer, whereas an experimenter will more likely monitor the activities.
- Experimenters may be less familiar with simulator operations than instructors typically would be.
- Both experimenters and instructors require a clear view of the pilot activities at all times.

As a result, the ECS requirements should also include:

- reconfigurability in software and hardware
- transparency of the experiment under investigation and operations of the simulator
- proximity to the pilots, and direct optical path to the control manipulators

Experimenter Activities:

As a result of the aforementioned requirements, the ECS facilities will include the following:

Master ECS (off-board, located in a glass cabin adjacent to the simulator bay)

- Configuration of the ECS environment
- Communicating with persons on board
- Data post-processing

Slave ECS or Master ECS (depending on configuration selected by Master ECS)

- Start of simulation session
- Monitoring simulator state
- Monitoring and interacting with flight displays and controls

Slave ECS (on-board)

- Monitoring pilot activities

While both an off-board and on-board facility will be available, the remainder of the text will focus on the on-board ECS and the displays which are common at both of these stations.

ECS Displays

As a result of the above activities, three displays will be available to the experimenter:

Main display

- Simulator status, configuration and systems warnings
- Flight vehicle status and configuration
- Simulation environment
- Experiment data (from a pre-established experiment plan)
- On-screen presentation of (pre-)selected variables, or plots from previous flight trials (overlays)
- Flexible labelling system for logging of runs

Map display

geographical data regarding the flight vehicle with respect to its environment

Auxiliary Display

Direct presentation of flight variables can often be facilitated by presenting the experimenter with an echoed image of a flight instrument (Primary Flight/Navigation, EICAS, CDU, MCP, etc.). A third display can be provided for this type of information. Additionally, it may be of value to provide the experimenter (particularly an off-board operator) with a live video image of the pilots from a rear-facing camera. These features, if allowed by interior volume, could be provided by a third display.

Display Functionality and Hierarchical Structure

Because of the multitude of functions necessary for the ECS displays, and since not every function is available in every system mode, the functionality is indicated per system mode. This was determined by conducting four "walk-through" experiments, with individuals typical of future users. The participants were asked to carry out a fictitious experiment, and then describe what functions they would use, and in which order. By examining the results, a proposed hierarchy in all functions, with respect to the expected importance, and also the frequency of use of each *category* of functions, could be realized.
The most important functions which require alteration during the real-time experiment were found to be as follows:

(a) mode transitions
(b) functions for managing the logging and presentation of experiment data
(c) "help" and "undo" -type functions
(d) functions for controlling SRS subsystems
(e) functions for controlling the aircraft variables

This research indicates that, although intervention by the experimenter may be necessary, the higher importance of observation dictates the ECS functionality.

Prototype ECS Displays

Two different concepts of the *main-display* have been developed[6]. Experiments with a user population resulted in the findings are listed below:

- system unfamiliarity does not influence an individual's ability to correctly determine the mode or corresponding transitions
- for reasons of clarity and efficiency, users tend to favor *single-layer menus* (main menu only) above *multi-layer menus* (main menu with submenus)
- graphically represented information is understood much quicker and better than in tabular form
- searching the programme for the right function consumes significant time

A demonstration program for the ECS main-display was developed, and controlled via a touch-pad as the cursor control device. The main conclusions are the following:

- Users appear to learn to handle the touchpad correctly within a few minutes, but further research on the use of touchpads in the high dynamic environment of the SRS is still necessary
- Although most subjects are able to control the mode system correctly without a separately-displayed mode scheme, subjects appear to operate the mode transitions quicker and with less mistakes when presented such a scheme
- The use of aviation terminology has little impact on an unfamiliar individual's ability to find necessary functions
- Confusing function terminology should be avoided

Display Menus

From the above research, the function categories are given the following names:

1. Logbook:
   system login functions and *project specification*
2. Data Control research data handling: logging and presentation
3. Set Subsystems SRS-subsystem handling: *enabling & disabling, control loading models*
4. Set Aircraft:
   simulated aircraft handling: *type, position, weight, etc...*
5. Set Atmosphere:
   simulated atmosphere handling: *winds, clouds, effects, etc...*
6. Set Runways:
   runways handling: *length, lights, etc...*
7. Set Hazard Vehicles:
   simulated hazard vehicles handlig: *amount, time-to-impact, etc...*
8. Set Radionavigation Systems:
   simulated radionavigation handling
9. Locks & Freezes:
   simulation control: *locks, freezes, snapshots, etc...*
10. Scenario Setup:
    specification of *manual, time-* and *event-driven scenarios*

Packaging the On-Board ECS

A consequence of mass-reduction and the lowering of the flight-deck structure below the motion system upper support gimbals is that the internal volume is restricted. The ECS design therefore requires consideration of all requirements, and selection of the priorities. Neither the volume of the ECS, nor the mass of related equipment should adversely influence the utility (and ease of use) of the simulator, or its performance.

The maximum effectiveness of the ECS could be achieved by providing a forward-facing seat, with instrumentation within easy reach. The displays are aligned at 45 degrees from the simulator centre-line, and do not block the view of the pilot controls. The proximity to the pilots further allows the experimenter full view of the outside-world display.

Due to severe volume penalties of CRT displays in the restricted space of the ECS, flat-panel TFT displays were selected. Proximity to these screens would ease readability. An analysis of display control devices (mouse, touch-screen, touch-pad, track-ball) led to the selection of a touch-pad, due to the ease of its use in a dynamic environment[7]. The palm support, integrated into the seat design, stabilizes the hand. Adjacent to the touch-pad is a numerical keypad, for rapid data input (Figure 3).

The on-board ECS hardware is as follows:

1. Experimenter Seat, with integrated touch-pad cursor control
2. Three 14-inch TFT colour displays, supported by rotating horizontal arms
3. Control panel with mechanical controls of:
- Emergency Stop
- Communication with off-board (master) monitoring station
- Communication with pilots
- Lighting
- Air conditioning of interior

4. Work table (retractable)
5. Document storage

The ECS seat and displays can be moved with ease to allow clear access between the pilot stations and the exit door of the shuttle.

ECS Seat Design

An analysis of the ergonomic aspects of current pilot and instructor seats revealed that these designs barely meet the ergonomic demands of the SIMONA ECS. Moreover, pilot seats (which are often used as instructor seats in commercial training simulators) are rated for accelerations up to 16 g, which is far beyond the simulator's 5 g design limit. Altering aircraft seats is also a difficult task; therefore, SIMONA elected to design its own ECS seat[8]. This would be moveable (fore and aft), and allow a wide population to make use of the seat while being able to observe the outside-world display.

The ECS seat is shown in Figure 4, and will be constructed of aluminum tubing and sheet. Fore and aft movement is achieved by rail-mounting the seat assembly, attached through a telescoping central column. The vertical movement is electrically-driven. Rotation about the vertical axis is also provided. The backrest and seat subtend a constant angle of 95 degrees. This is maintained when the seat is tilted back to a maximum of 10 degrees. The backrest integrates also the headrest and a lumbar support. The seat is provided with a four-point safety harness.

ECS and IOS Research Opportunities

One aim of the SIMONA Research Simulator is to develop through experimental evaluations the requirements for future training systems. This will include investigations into the exchange of information between the trainer and the training environment, and the ergonomics of the instructor station and instructional/monitoring tasks. Research will extend to non-standard training methods, making use of remote instructor station concepts. Automation in the instruction task could be evaluated, as well as the effectiveness of the instructor in monitoring multiple training tasks simultaneously.

In the process of such research, one could foresee that the *instructor* would serve as the experimental subject, and be evaluated on his/her ability to act upon this. The environment of a full flight simulator would appear to offer unique opportunities in this type of research: The environment is representative of training sessions and, more importantly, it is possible to present various configurations of hardware and software.

Off-Board and Remote Instruction

The fundamental aspects of off-board instruction can be investigated in the SIMONA research centre. Adjacent to the computer facility is a room dedicated for such research, as well as for pilot (de-)briefing. The remote ECS could be used to indicate the advantages and difficulties of placing the instructor outside the simulation area.

Research Into Future Training Scenarios

Training scenarios, particularly in commercial civil aviation, have focused on developing the skills of the crew within the simulated aircraft. Attention focuses on the ability of the crew to manage the flight through a set of conditions imposed by the atmospheric environment, and the aircraft systems. Recently, the introduction of Traffic Collision Avoidance Systems has exposed simulator crews to interactions with other vehicles, however on a limited scale.

Traffic growth forecasts indicate that flight operations will require significant interactions between aircraft, coordinated through Air Traffic Management systems. High traffic density will necessitate flight crews to be prepared for aversions, operation in close proximity to other aircraft (of various sizes), and rapid changes to flight plans. The simulator would appear to provide an interesting opportunity to investigate these situations, if these scenarios can be realistically simulated. While all traffic is coordinated by ATM, the high-density scenarios could often require crews to be prepared for increased interaction between aircraft.

Experiments into the effectiveness of interaction by off-board persons, utilizing also an increased level of automation in the control of lesson plans and in monitoring performance, could have significant benefits to training in the future. One could propose a scenario where a training centre is operated by fewer instructors who simultaneously monitor multiple simulators. This training concept is shown in Figure 5.

Research into the feasibility of off-board instruction and monitoring could lead to the development of a large-scale interactive training environment, networked between simulators and training centres. The inclusion of separate air-traffic controllers could provide a more realistic interaction between aircraft and their environment. Training could extend to the operators of with special functions such as cabin crew, and with representative of air traffic situations today. The applications of this type of situation, cost permitting, would provide information to airport operators and developers, and in general the air transport community.

Conclusions

This paper is intended to highlight the benefits of research into experiment control and instructional functions in flight simulation. The latter can clearly benefit from the former, if the environment allows the research results to be applied in the training environment. For these reasons, SIMONA has elected to demonstrate the applications of new techniques in display presentation, instructor interface design, instructor station ergonomics, and comfort. The SIMONA Research Simulator will soon enter operation as a research tool and also a platform for new simulation and training concepts. In all of these activities, the role of the human in controlling the experiments will be of high importance. This inter-activity will be feasible through an effective ECS environment.

References

[1] Advani, S.K., "The Development of SIMONA: A Simulator Facility for Advanced Research into Simulation Techniques,

Motion System Control and Navigation Systems Technologies". AIAA-93-3574-CP. From AIAA Flight Simulation Technologies Conference, Monterey, August, 1993.

[2] Advani, S.K., Tooren, M. van, Winter, S. de., "The Design of a High-Performance All-Composite Flight Simulator Motion Platform". AIAA-95-CP. From AIAA Flight Simulation Technologies Conference, Baltimore, MD, August, 1995.

[3] Advani, S.K., and Verbeek, R.J., "The Influence of Platform Inertial Properties on Simulator Motion System Performance". AIAA-94-3418-CP. From AIAA Flight Simulation Technologies Conference, Scottsdale, August, 1994.

[4] Advani, S.K. and Mulder, J.A., "Achieving High-Fidelity Motion Cues in Flight Simulation". From Proceedings of the AGARD Flight Vehicle Integration Panel Symposium 'Flight Simulation - Where are the Challenges'. Braunschweig, Germany, May 1995.

[5] Gool, P.C.A. van, "SIMONA Software Requirements Specification". Internal Report, SIMONA, 1995.

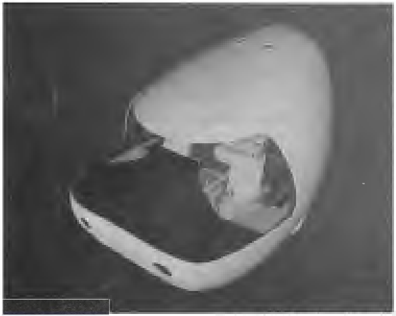
[6] Geest, B. van der, "User Interface of the SIMONA Research Simulator Experiment Control Station" (in Dutch). Graduation thesis report, Delft University of Technology, Faculty of Industrial Design Engineering, April 1996.

[7] Crane, J.M., Bang, E.S., and Hartel, M.C., "Standardizing Interactive Display Functions on the 777 Flight-Deck". Boeing Commercial Airplane Group, SAE Technical Paper Series 942093, October 1994

[8] Wattimury, A.A., "Experiment Control Station of the SIMONA Research Simulator" (in Dutch). Graduation thesis report, Delft University of Technology, Faculty of Industrial Design Engineering, January 1996.

Activity	**Activity Load** (Percentage of total time)
Observing the pilots' procedures	46
Monitoring IOS screens	13
Interacting with IOS controls	10
Interacting with IOS controls while observing pilots	9
Interacting with IOS controls while reading	3
reading	8
writing	7
general	4

Table 1. Instructor operator activities and workload observed during LOFT training sessions



[illegible]

Figure 1. SIMONA Research Simulator Flight-Deck "Shuttle"

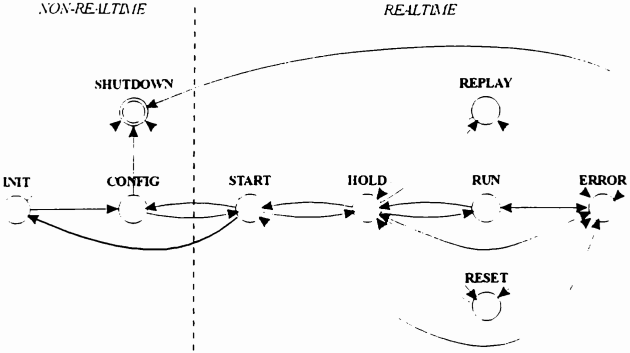


Figure 2. SIMONA Software structure (currently under development), showing states and mode trainsitions

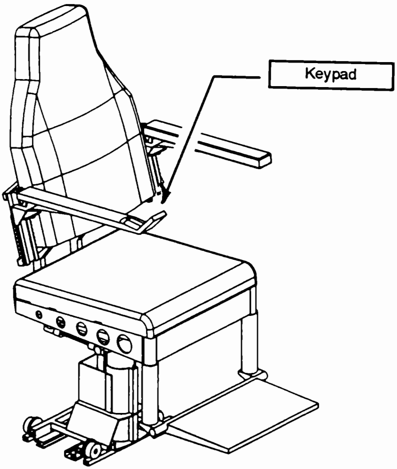


Figure 3. ECS seat, showing location of integrated numerical keypad

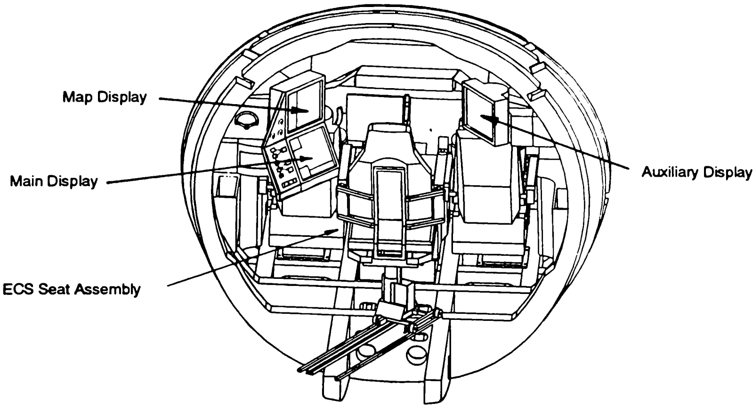


Figure 4._SIMONA ECS seat and working environment, viewed from aft of SHUTTLE

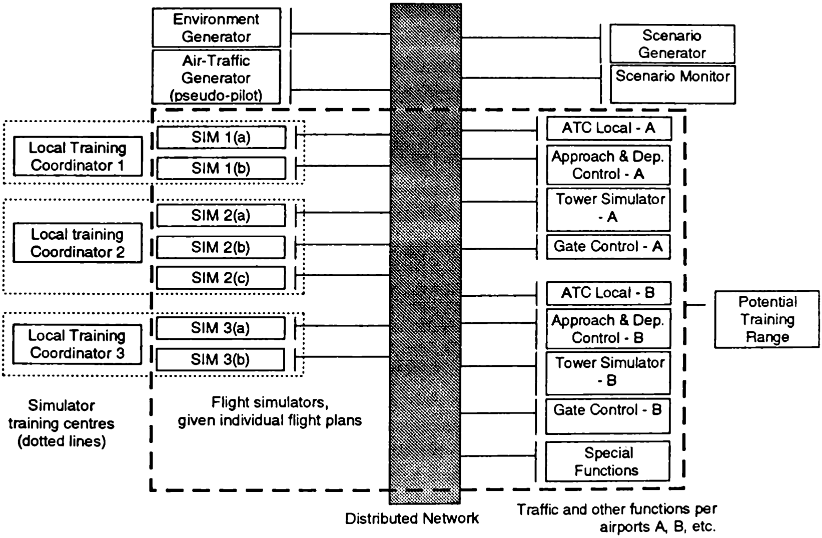

Environment Generator
Air-Traffic Generator (pseudo-pilot)
Scenario Generator
Scenario Monitor
Local Training Coordinator 1
SIM 1(a)
SIM 1(b)
Local training Coordinator 2
SIM 2(a)
SIM 2(b)
SIM 2(c)
Local Training Coordinator 3
SIM 3(a)
SIM 3(b)
Simulator training centres (dotted lines)
Flight simulators, given individual flight plans
ATC Local - A
Approach & Dep. Control - A
Tower Simulator - A
Gate Control - A
ATC Local - B
Approach & Dep. Control - B
Tower Simulator - B
Gate Control - B
Special Functions
Potential Training Range
Distributed Network
Traffic and other functions per airports A, B, etc.

Figure 5. Commercial flight training scenario employing remote off-board instructor stations. Flight and ATC simulators are networked in a common environment.

AIAA-96-3510-CP

DIAGRAM-BASED AUTO-GENERATION OF SIMULATION SOURCE CODE IN THE BOEING AIRPLANE SYSTEMS LABORATORY

K.C. Babb
BCAG Airplane Systems Laboratory
Simulation Design and Integration
M/S 19-MH, P.O. Box 3707
Seattle, WA 98124

Abstract

The Graphical Simulation Development System (GSDS) is an internally-developed toolset used for the design, documentation, and automated coding of aircraft subsystem and LRU (Line Replaceable Unit) simulations within the BCA (Boeing Commercial Airplanes) Airplane Systems Laboratory organization. Its primary features include a customized graphical diagram editor, a multi-user database maintenance facility for data dictionaries and diagram/module information, and a code generation capability for automatic creation of HOL (high-order-language) source code modules which implement the algorithms and inter-diagram hierarchical references depicted in the user's system of diagrams.

Diagrams may be analyzed and verified from within the graphical editor, with checking performed on completeness, syntax, input/output type compatibility, validity of feedback loop constructs, past-value usages, initialization requirements, and module parameter connection requirements (dataflow). In order to facilitate quicker maintenance of diagrammed systems, a dataflow diagram for a given hierarchical module may be auto-generated using the i/o information for its subordinates, once the control flow diagram for the same hierarchical module has been completed.

Variable dictionaries are supported through extensive maintenance features, and changes made to variable definitions can be automatically incorporated into referencing diagrams without manual user effort. For example, a variable may be renamed in the dictionary and (at user request) all references to the old name found in existing diagrams can be automatically changed to the new name.

The code generation utility within the system converts the information within the diagrams and dictionaries into directly compilable standard Fortran, Ada, or Pascal source code, and features an assortment of user options for control of code configuration (subroutine/procedure, function, program, or in-line) and certain optimization capabilities. Declaration files for needed variables and common blocks, records, and/or packages are also generated. User options are available to control target file location and naming, as well as language extension features such as end-of-line comment delimiters in Fortran.

Diagram information consists of graphical attributes such as symbol positioning, sizing, and connection routing, as well as semantic details regarding direction and content of data flow and control flow, variable references, and hierarchical relationships between other diagrams.

Options within the system permit reconfiguration of diagrams or sets thereof via "layering", in which diagram components assigned to particular overlay designations may be designated active or inactive. This yields easy switching between viewing and coding of configurations intended for different types of simulations, or for simulation code versus embedded code.

Translation utilities in GSDS provide for filtering of diagrams to analysis tools such as Boeing's Easy5W®, and the commercial product Software Through Pictures®, as well as graphics filtering to PostScript®, HPGL® and the Interleaf® word processor's ASCII markup language. Documentation and report generation features allow for diagram and dictionary information to be automatically presented in special formats for publication purposes, without manual alterations by the user.

The graphical portion of the user interface is implemented in X Windows® and Motif®, and provides components for performing editing, variable and module definition, diagramming, analysis, translation, and code generation.

A comparison case study examining the time required to diagram and code specific algorithms, as performed by multiple engineers with varying levels of toolset familiarity, has shown a time savings average of approximately 50% for new module creation. The true value of this approach, however, is found as updates and changes are made to the design. Each diagram can be

changed and its code regenerated in a matter of minutes, and the attendant formal documentation updates to diagrams, dataflow information, and data definitions are automatically available on demand.

Introduction

The use of tools which generate compile-ready source code to implement the algorithms and modular relationships depicted in a set of functional and hierarchical block diagrams has been demonstrated to be of considerable value in a variety of simulation environments. Boeing has developed different types of software which support aspects of this process, according to requirements specified by simulation developers and their customers. This discussion presents the development and usage aspects of a software package developed for use in a simulation laboratory environment.

The Graphical Simulation Development System (GSDS) is primarily used for the design, documentation, and automated coding of aircraft subsystem and LRU simulations. The use of such capabilities dates back to the early 1980's and the 767 program, which spawned the original one–diagram–one–subroutine tool upon which GSDS is based. The toolset's current primary features include a customized X Windows/Motif–based diagram editing and analysis utility, a multi–user database maintenance facility underlying all of the major tools (used for data dictionaries and diagram/module information), and code generation capabilities providing automatic creation of high–order–language source code files. The generated code consists of user–configured and/or standard commentary and header information, revision control system information, statements which declare and initialize variables, and the modules and in–line fragments which implement each of the algorithms and inter–process hierarchical references depicted in the user's system of diagrams. Additional toolset features include a source code comparator which is used to identify differences in functionality between versions of a particular module. Various secondary and administrational capabilites are also supported, such as those which facilitate use of typical revision control sofrware to provide configuration management for both the system definition and the generated code.

The features of the toolset have almost entirely been user–driven, through requests submitted by customers and prioritized by a change control board consisting of user group representatives. The number of active users both internal and external to the organization has typically been between 100 and 150, and they have been encouraged to convey their specific needs to the development team. This has led to the incorporation of certain unique capabilities, some of which have proven of limited actual use while others have contributed greatly to easing the process of creating and documenting simulations. Both successful and less–than–ideal usage and development experiences are discussed herein.

Functional Boundaries

The end–result functional boundary of the GSDS capabilities is the production of source code for the simulation, and its accompanying documentation; at the time of the toolset's development in ASL, an environment was already in place for linking, executing, controlling, and displaying simulations and analyzing their data. It was thus deemed unnecessary to provide redundant capabilities within GSDS itself. Figure 1 depicts the high–level process environment in which the toolset is typically used within ASL organizations. The shaded processes and dashed dataflows are those which do not fall within the domain of GSDS usage.

GSDS features numerous interface capabilities which support use of the generated code within the existing simulation environment. These include auto–generation of "shortcut" aliases for variables (to minimize user effort required to access them within the simulation), as well as monitoring system display page definitions containing diagrams and display elements for input, output, and test variables. These display pages set up window elements used to monitor simulation variables via graphical and textual feedback, with the diagram shown on the page and active display elements located at the positions of the variable names on the diagram. Once refined according to specific utilization experiences, these capabilities have proven valuable to users once their systems have become ready for testing in the simulation environment.

Simulation Definition Component Overview

The definition of a simulation within the toolset includes the diagrams which depict algorithms and hierarchical references within the defined system. Each diagram depicts an entity known as a "process", which can correspond to a function, procedure, main program, or an in–line code fragment. Processes are identified by their unique names, which are used in the generated code and thus must comply with the least flexible of the generated language procedure name constraints (currently imposed by Fortran). This requirement exists to facilitate easy transition from one generated language to another, a fundamental constraint upon the toolset from its earliest days.

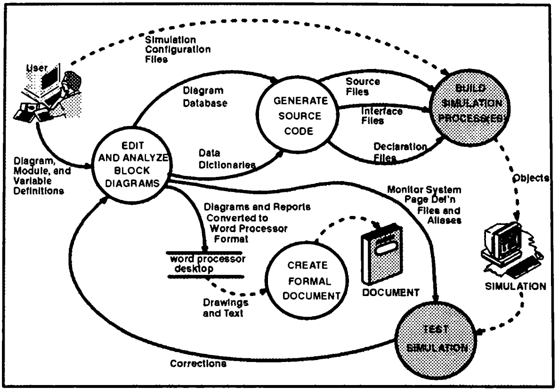

Simulation Configuration Files
User
Diagram Database
GENERATE SOURCE CODE
Source Files
Interface Files
BUILD SIMULATION PROCESS(ES)
Diagram, Module, and Variable Definitions
EDIT AND ANALYZE BLOCK DIAGRAMS
Data Dictionaries
Declaration Files
Objects
Monitor System Page Def'n Files and Aliases
Diagrams and Reports Converted to Word Processor Format
word processor desktop
CREATE FORMAL DOCUMENT
DOCUMENT
SIMULATION
Drawings and Text
TEST SIMULATION
Corrections

Figure 1 Simulation Development Process

Diagrams which depict the data manipulation aspects of a system at the lowest level, including the simple mathematical and logical operations and usage of basic elements such as integrators and filters, are referred to as "primitives". These diagrams are also known as basic or functional block diagrams (BD's). Above the BD's in a simulation depiction are "non-primitive" hierarchical diagrams which show execution sequences and conditions (control flow) as well as the interchange of data items between lower-level diagrams (data flow). These are known as flowcharts (FC's) and dataflow diagrams (DD's), respectively. A complete simulation definition will consist of all three diagram types.

The generation of code for a defined and successfully analyzed simulation definition tree of diagrams hinges upon the control flow information for the non-primitive processes, and the operational information for the primitives. Thus the hierarchical dataflow diagram is not strictly required for code generation and preliminary checkout, and the portion of the automated analysis which verifies the input and output compatibility and interface completeness of dataflow diagrams and their sub-processes may be temporarily disabled by the user during early development. The toolset includes features which can auto-generate the dataflow diagrams for non-primitive processes, once all subordinates for each one have been completed. This feature simply examines the input and output requirements for each subordinate, establishes process symbols representing the lower-level diagrams, and makes connections between those having i/o's of the same data name. In addition, the DD creation facility provides external input and output symbols for those data items which do not connect internally between subordinates. An auto-created dataflow diagram of this type will by definition pass the system dataflow analysis checks, since they are based upon the identical criteria used in the diagram's creation. The auto-layout approach for creating these diagrams incorporates top-down, left-right sequencing of the symbols representing subordinate diagrams (based upon their order of execution as shown in the flow control diagram), and the auto-routing of connections attempts to minimize crossings, overlaps, and the use of corners. Users have tended to post-process such auto-created diagrams in order to rearrange the layouts according to their own aesthetic preferences; this practice has somewhat limited the potential time savings offered by this feature.

Symbology

At the non-primitive level, few symbol types are required since the purpose of each such diagram is simply to depict the data and control flow relationships between diagrams beneath it in the system definition structure. Symbols in flowcharts accommodate decisions, loops, references to lower-level diagrams, and standard terminators (BEGIN, END, etc.). Symbols in dataflow diagrams depict input and output names for individual data items and groups thereof, and the subordinate diagram references. In addition, the user may select a diagram appearance protocol which causes a dataflow or flowchart symbol representing a lower–level diagram to be configured according to whether the lower–level entity is nonprimitive, primitive, or undefined.

At the primitive level, a fixed set of simple operational symbols is combined with a user–configurable set of function symbols. The operational symbols provide familiar methods of depicting inputs and outputs of varying categories, including globals and argument (parameter) list items. They also provide many mathematical operations such as addition, subtraction, and exponentiation; logical functions such as AND, OR, XOR, and latches; and simple control flow via single–assignment and compound switches. The function symbols allow unlimited references to components such as integrator and filter functions which may reside either in user or standard libraries, or within the simulation definition itself as primitive diagrams which are coded as functions or procedures. Employing a certain level of fixed, standard symbology has proven to be of value since it minimizes the use of obscure or custom symbols for basic operations, which may not be familiar to a wide spectrum of users. Interface features and requirements of the function symbols are largely user–configurable via ASCII function interface definition dictionaries, as are the graphical parameters of each function icon such as symbol size, pictorial contents, and textual labels.

Variable Definition Dictionaries

Variables are managed via a scheme which incorporates user–controlled attributes including common block/record affiliations, Fortran equivalences, user type definitions (for use with Ada), INCLUDE file assignments, and hierarchical container data for variables whose dataflow aspects are managed in groups known as vectors.

A variable is indexed by its user–entered name, and its definition is located in a database referred to as a Data Dictionary. Each such dictionary may be implemented either as a multi–user Interbase® database, or as a readable and editable ASCII file. The latter are often used in environments within which only one user will be accessing a set of definitions, and/or under circumstances which require exchanging of dictionaries among varying users and even across varying platforms. One benefit of the ASCII dictionary format is the ability to employ standard configuration control tools such as SCCS to directly manage the dictionary files. A user's system environment may include as few or as many such dictionaries as needed. A multi–user dictionary may be readily converted to and from its ASCII equivalent in order to facilitate ready use of both formats and the advantages of each.

Each variable definition within a dictionary contains attributes such as name, data type, initial value (or a list thereof for arrays), common/record name, range, units, a textual description, and lists of diagrams which either consume or produce the variable. Any combination of these and other attributes may be printed in a formatted output using the dictionary report features.

If a variable entry is renamed by the user, an option is provided to search the entire system definition environment for references to that variable which are contained within diagrams, and to automatically substitute the new name for the old one within those references.

Documentation Support

The user communities which shaped the initial GSDS requirements envisioned the documentation portion of the simulation development process as a primary need rather than an afterthought. It was believed from the early stages that a primary benefit inherent in the toolset would be its use for creation of diagrams to appear in the SCD's (Specification Control Drawings), an essential and highly formal component of the simulation documentation suite which is provided to LRU vendors as their formal design specification. Thus the point of control for both simulation code and the documentation which must accurately depict it would be the same; the GSDS model definition. As a result, the toolset provides substantial support for cosmetic control of diagram contents, and direct output to the word processing utility of choice within the organization.

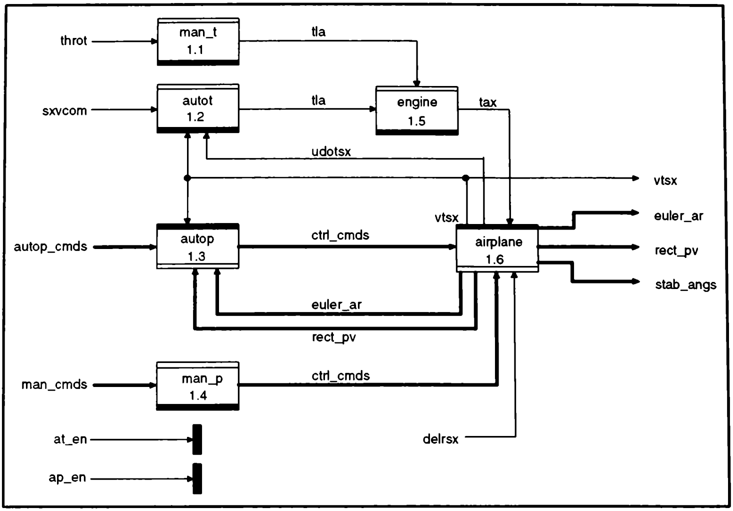


Figure 2 GSDS Dataflow Diagram Showing Simple Names

A single GSDS diagram can take multiple forms based upon user–selected settings. Process reference symbols can be labelled with the actual name of the process (which is used in generated code), the textual definition of the process (which appears as comment lines in the generated code), or an alias for the process name, permitting the use of mixed–case descriptive names while avoiding potential confusion caused by erroneous mixed–case database entries. Variable reference symbols have similar options. Symbols may be sized by the user as needed to enclose lengthy text descriptions. Figures 2 and 3 show the same diagram printed with different sets of options; the first contains only simple variable and process names and the default symbol sizes; the second shows variable and process descriptions and user–selected symbol sizes. The only manual step required to switch between the two views is the setting of two options within the diagram editor user interface.

In addition, output capabilities are supported for PostScript and HPGL, to meet the needs of users wishing to print diagrams directly without using an intervening word processor, to import the diagrams to other tools, or to create display media—such as J–size composite diagrams—encompassing large–scale systems or subsystems. The HPGL capability, combined with an automated composite–diagram creation feature, allows nearly complete automation of a process formerly performed by printing dozens of small diagrams and pasting them together manually to obtain a large–scale, fine–detail overview of a system subtree.

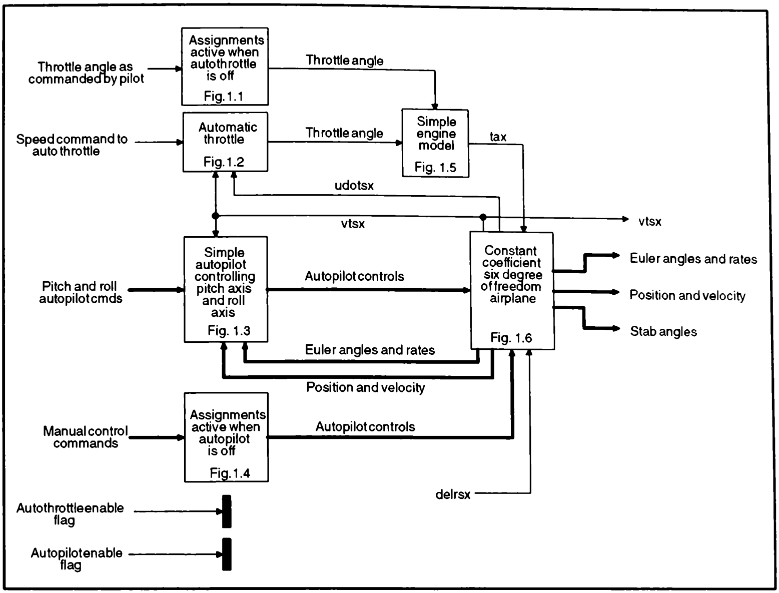


Figure 3 GSDS Dataflow Diagram Showing Description Information for Documentation

Report generation features within the toolset can provide textual output showing a wide variety of information regarding diagrams and their relationships as well as variables. The entire simulation definition can be depicted in a hierarchically-formatted tree listing, indended by hierarchical reference level. Variable information which can be reported includes all attributes of the dictionary definition (type, intial value, range, units, etc.), as well as the names of all producer and consumer processes througout the system. This latter feature has proven valuable both for analysis and documentation purposes, for easy identification of diagrams and modules in which a particular variable is used and/or set.

Each diagram's position in the hierarchical tree corresponds to a user–configurable figure number, which may be automatically included in all diagram outputs and references for documentation purposes. This number will also appear in comment lines within the source code, accompanying the process name and description. The top–level process of a system (which may correspond to the main program, if the user so desires) is typically numbered 1 (though the user can manipulate this as needed). The next level of processes is then numbered 1.1, 1.2, etc., with the subsequent level being 1.1.1, 1.1.2, 1.2.1, and so forth. These numbers are automatically maintained by the dictionary interaction features of the toolset, though the user may assign different numbers within each level if desired.

Diagram Analysis

The analysis of primitive–level diagrams in GSDS is performed for attributes such as interface verification, type checking, parameter connection completeness, and coding order validity (including a check for invalid loop constructs). Error conditions result when configurations are detected which prevent successful code generation. Warnings result when conditions are encountered which may yield unintended or questionable results. Certain error/warning conditions are dependent upon the language chosen for code generation, since the type compatibility requirements vary across the languages. An example of a situation of this nature would be the mixing of integer and real quantities as inputs to a summing junction, which is permitted for Fortran but not for Ada. The diagram analysis features take into account such variations among language requirements.

A textual error/warning log, output to a file and accessed in the diagram editor via a standard Motif file viewer, explains the nature of each error and identifies the associated symbol(s) via unique identification numbers. These numbers may be displayed adjacent to their symbols in the diagram editor (and on any hardcopy, if desired) to further speed the user task of determining which part of a diagram may be involved in a particular error condition.

In the diagram editor the analysis provides a graphical form of feedback in addition to the textual and log file output; each symbol and/or connection which generates an error condition is highlighted with a shaded background, while each component which generates a warning condition is highlighted via bolding of lines and text.

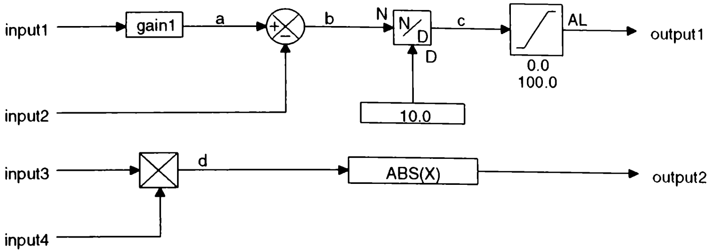
Code Configurability

In order to facilitate the use of generated code on multiple simulation host platforms, the toolset output code (specifically the Fortran) has historically been required to be configurable for different variations of compiler features and language extensions. For example, the early Harris simulation hosts employed a compiler which required a "$ADD" syntax instead of the more conventional "%INCLUDE", and the end–of–line comment delimiter character has varied from "!" to "{". User–controllable option settings have been provided to address needs such as these. The code generated by GSDS does not employ any language extensions or non–standard constructs unless specified by the user with options of this nature.

Compliance with coding standards is supported by capabilities which either allow the user to specify a text file containing comment lines to be used as a preamble in the generated module. In addition, the header format for each generated source file includes the process name and hierarchical number information, process description text, lists of user–defined local variables and auto–generated local variables, and lists of external inputs and outputs.

Inputs and outputs to a process depicted in a diagram can be identified in multiple ways, each of which results in a different form for the means by which a particular data item interfaces to the code for the diagram. An input or output may be designated as a global entity or a local one, and locals may be configured such that they are call list parameters (if the diagram is to be coded as a function or procedure), or they may be designated as being local to the module containing the reference to the diagram (if it is to be coded in–line to one or more higher–level processes).

The order in which code is generated for the parts of a diagram may be controlled by the user simply via recognition that the coding order is output–driven and top–down–left–right in nature. This means that the path to each overall output is traced back through upstream symbols and intermediate variables until the overall inputs to the diagram are reached. The expression for the furthest–upstream computation is coded first, then the next one downstream, then the next, and so on until the assignment statement for the output is reached. This is done for each of the overall outputs, beginning with the one with the uppermost position in the diagram. Figure 4 shows an example of a simple diagram and its attendant code, to illustrate the coding order protocol.



Generated code:

```
a = input * gain1
b = a – input2
c = b / 10.0
output1 = alimit (c, 0.0, 100.0)
d = input3 * input4
output2 = abs(d)
```

Figure 4 Simple Diagram and Coding Order Example

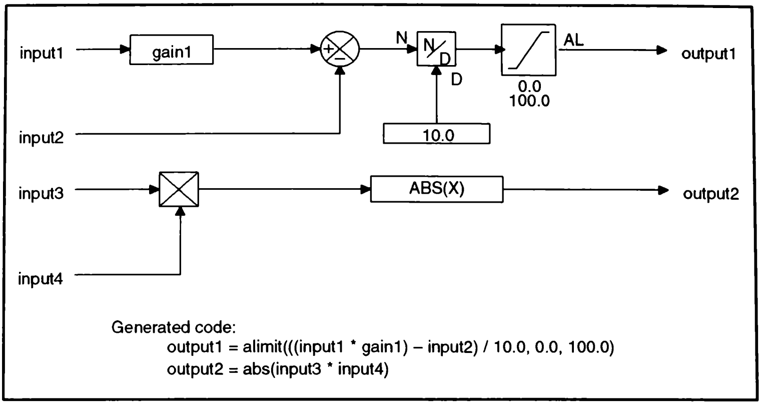
Optimization

Because of the need to keep simulation frame times down despite increasing levels of complexity, the toolset provides features which support the creation of faster, smaller code when it is reasonably possible to do so. One means of doing this is to "carry" expressions; the input of an operator may consist of the expression depicting the output of an upstream operator. An example of this is shown in Figure 5, which consists of the same diagram shown in Figure 4 without the intermediate connection names. The user may selectively control the occurrence of expression–carrying via assigning names to the connections between symbols; each such name causes the generation of a separate assignment statement.

An additional means of creating more efficient code is to provide for nesting of IF constructs; an expression or group of assignments which is upstream of cascaded switch symbols can be nested within multiple levels of IF blocks, according to options set by the user. Along similar lines, a symbol type is provided which allows multiple conditional assignments to be driven by a single flag; this permits a single IF construct to be coded, where the alternative (use of a single–assignment switch multiple times) would result in multiple IF constructs.

Language Feature Support

Support exists within the toolset for the user to create and employ user–defined types (for use in Ada), including subtypes, derived types, record types, arrays of records, and record types containing any or all of these, as well as "standard" types. Additional Ada language features supported include representation clauses for enumerations, records, and size and storage location attributes. The standard Fortran data types (integer, real, logical, character, etc.), and their equivalents in the other languages, are supported as well.



```
Generated code:
        output1 = alimit(((input1 * gain1) – input2) / 10.0, 0.0, 100.0)
        output2 = abs(input3 * input4)
```

Figure 5 Simple Diagram and Compacted Code Example

Operational Features and Usage Patterns

Interactive capabilities are provided by the Block Diagram Editor and the Dictionary Editor. The features supported in these tools—particularly the diagram editor—are directed at assisting the user with the tasks of diagram creation, diagram analysis and incremental code generation, data dictionary entry creation and maintenance, and dozens of other functions required to create and update a simulation definition. Many convenience features and shortcuts exist for speeding the initial drawing of the diagram; these include features which provide for repeating the last symbol type added with a single mouse button click, replacing a connected symbol with one of another type in one step (rather than the several involved in deleting it and then adding the new type and re–adding the connections), and one–time or repeated duplicating of any symbol or region within a diagram without impacting the contents of the cut/paste buffer (sometimes referred to as the clipboard). In addition, the diagram editor provides complete interactive code generation capabilities via which the user may analyze a diagram, and generate its code for examination while continuing to view and edit the diagram's contents. Capabilities of this type have been developed in response to user survey results indicating features which have the greatest impact.

One of the idealistic early goals of the toolset was that there would be no need for users to concern themselves with the details of the code (much like programmers employing high–order languages do not typically bother to examine the generated assembly language or machine code resulting from compilation). However, a reality of usage quickly developed which indicated that a large proportion of users felt more comfortable with the tools and with autocoding in general if they could quickly and easily generate and examine the generated source code, particularly during the initial diagram development phase. For this reason, GSDS provides features which exist solely to make the generated code more readable. Among these are identification of the beginning and ending points of the code for each diagram (useful when a diagram's code is in–line within a module containing code for multiple diagrams), user control of indentation, and user control of variable names (and of protocols used to generate temporary variable names if they are needed). This last feature was also considered a requirement for simulation monitoring and debugging, and has proven useful for both documentation and analysis purposes.

Each of the major tools within the GSDS toolset may be invoked separately from a command line or from another tool. It has been observed that once a simulation has been defined, many users employ various features of the toolset in "batch" fashion. The typical practice is to construct invocation scripts which invoke one or more of the tools with the desired setup options, and to run those scripts in the background. Among the functions typically performed in this manner are: (1) conversions of large numbers of diagrams and their definitions to pure ASCII files for CM and archiving purposes, (2) queueing of many diagrams for printing or filtering to a word processing utility or PostScript, and (3) generation of source code for an entire system definition or a large subtree thereof. For batch usage, the primary GSDS tools may be invoked with an unlimited list of diagram names which are to be processed. This enables, for example, a complete system definition tree to be queued for printing with a single command and just one opening and initialization of each utilized database. Usage tracking data indicates that approximately 25% of diagram editor invocations are performed for such purposes.

Simulation Usage Versus LRU Usage

Certain subsets of the GSDS user community have chosen to employ the generated code both for simulations and for testing purposes in prototype LRU's. The differences between these types of usage have led to the incorporation of features which permit quick reconfiguration of diagrams and omission of certain code items pertaining specifically to simulation usage, such as those for variable initialization based upon simulation mode. New capabilities are currently in work to support this type of use, including user-transparent (and user-controllable) support for fixed-point data types and arithmetic.

Test Case Development Time Comparison

In order to ascertain the order of magnitude of time savings which can be provided by coding a particular algorithm with GSDS versus coding it by hand, small test cases were created and implemented by engineers having varied levels of experience with the toolset. The time required to complete the designated task included both the time to create a document-ready diagram (either with GSDS or a drawing utility such as Interleaf or Autocad®) and the time to create compile-ready code (including all necessary variable declarations) for the applicable modules. An example of one of the diagrams used is shown in Figure 6.

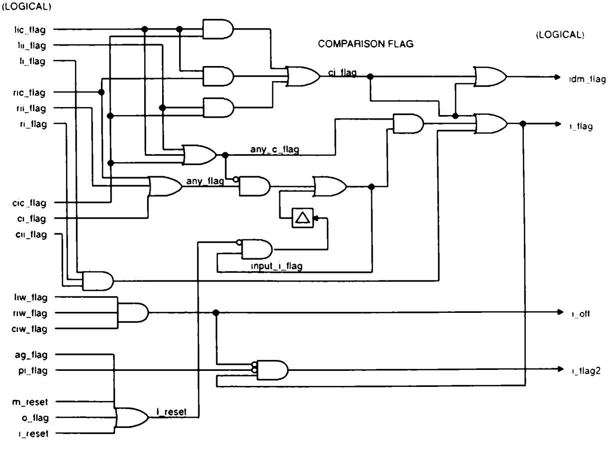


Figure 6 Logic Diagram Used In Development Time Comparison

The data gathered was as follows:

Digital Logic Diagram (as shown):
Average time to complete diagram and code in GSDS: **97 minutes** (minimum time 42 minutes, maximum time 130 minutes)
Average time to complete diagram and code manually: **192 minutes**

Simple Control Law:
Average time to complete diagram and code in GSDS: **57 minutes** (minimum time 38 minutes, maximum time 75 minutes)
Average time to complete diagram and code manually: **165 minutes**

While these examples are clearly much smaller than a typical development effort, it is believed that the results show that the use of a tool of this nature saves considerable time, particularly when the documentation aspects are thoroughly addressed. This advantage is in addition to those provided by the analysis features and simulation interface features which assist with verifying the correct functionality of the diagrammed algorithm and the code. Within both the initial creation and later maintenance stages, significant savings also are obtained via reduction of code verification efforts. A given diagramming construct will always generate the same code, eliminating the potential variations introduced by programmer interpretation, and if code and documentation updates are made via the toolset there is no need to repeatedly verify that the formal documentation and code are consistent.

Conclusions

It is clear from experiences both within this organization and throughout the industry that the use of auto–coding based upon graphical representations can yield time savings and improved quality throughout the simulation lifecycle, from formal documentation and design through long–term maintenance. Within the subject experiences with applications of this technology, the primary features of value to users have proven to be those which support quick, intuitive editing and analysis of diagrams, and those which provide readable, well–commented code which can be configured according to the user's priorities. In addition, the analysis capabilities have been found to be most useful if they provide prompt graphical and code preview feedback which can be immediately used to make modifications to any affected diagram(s). Automation of otherwise–tedious tasks, such as the propagation of renaming changes through a complete system definition or the duplication of components ranging from single symbols to entire subtrees, has also been of considerable value. The use of single–point–of–control database dictionaries as repositories of variable and dataflow information has proven advantageous for supporting both multi–user development and configuration management. Enhancements to the toolset forecast for the near future include additional reconfiguration and automation features, improved interfacing to the simulation execution environment, and an expanded set of available generated languages to support a broader spectrum of usage types.

AIAA-96-3512-CP

Integration of a MATRIXx Simulation Model into a Fixed-Base Aircraft Simulator

Peter R.W. Dietz
Canadair, Bombardier Inc.,
Montréal, Canada*

Steve Hébert
Canadair, Bombardier Inc.,
Montréal, Canada =

This paper describes the software development undertaken to integrate a simulation model produced with the MATRIXx suite of software into a fixed-base flight simulator. The software developed to facilitate this integration needed to be easy to use; be able to accomplish the integration of the simulation models with the simulator in a highly automated manner ; be able to initialize the simulation at flight conditions specified by the simulator test engineer; and the software had to provide the simulator user with the means of importing data recorded during simulator test sessions into applications that may be used to analyze the data. Achieving these requirements has produced a virtual prototyping capability that permits simulation models produced by design engineers using modern GUI tools to be easily and quickly installed on a fixed-base simulator. The design concepts represented by these simulation models may then be "flown" on the simulator by a test pilot. The data gathered during these simulator flights may be analyzed and used to refine the design concepts.

Introduction

The current trend in computer-aided design is toward "virtual prototyping", in which designs of hardware are formulated and tested entirely on computers before actual hardware is manufactured. Sophisticated simulation software with convenient graphical user interfaces may be used in the design phase to produce simulation models of the systems of interest. In the case of the aeronautical industry, the resulting models are then installed on a piloted flight simulator to be tested. The results of the simulator experiments are used to refine the initial design. The design-simulation process is iterated until a satisfactory system performance is obtained.

Figure 1 presents the hardware/software configuration that is desired. Workstations in the design departments are equipped with software for formulating new designs and turning these designs into a simulation model. A simulator is available that is properly equipped to permit pilot-in-the-loop simulations.

Communication between the two facilities are accomplished over a LAN network. To provide a means of installing the simulation models developed on the workstations into the simulator, and using data from the simulator on the workstations, a software integration process has to be developed and additional code processing and data processing software is required. This paper is concerned with both the software integration process and the code processing and data processing software suite.

The design tools of concern in this paper are Xmath™, SystemBuild™ and AutoCode® from Integrated Systems Inc. (ISI).= Xmath™ is a command driven interface for performing mathematical calculations. SystemBuild™ is a GUI tool used to model the system of interest in block diagram format. AutoCode® converts the block diagrams into C code. A model produced with these tools shall be referred to

*Engineering Specialist, Aerodynamics Simulation Group.

= Staff Engineer, Aerodynamics Simulation Group.

= Xmath and SystemBuild are trademarks of Integrated Systems Inc. of Sunnyvale, California. AutoCode and MATRIXx are registered trademarks of Integrated Systems, Inc.

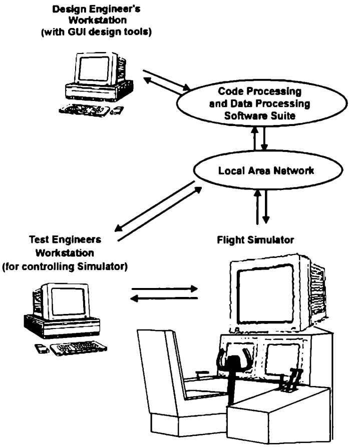

Design Engineer's Workstation (with GUI design tools)
Code Processing and Data Processing Software Suite
Local Area Network
Test Engineers Workstation (for controlling Simulator)
Flight Simulator

Figure 1, Schematic of Virtual Prototyping Hardware/Software Arrangement

here as a MATRIXx® model (which is the name of ISI's original version of this software suite).

The simulator referred to in this paper is the Reconfigurable Engineering Flight Simulator, or REFS. This is a fixed base simulator with electrically-powered flight controls which provide realistic force feedback to the pilot. The control force models are run on a PC. Interface between hardware and the models is provided by a second PC. A single channel visual system using a 37 inch monitor is positioned in front of the pilot position to provide an "out-the-window" display. Two 19 inch monitors are positioned in front of the pilot, below the out-the-window display monitor, to provide cockpit instrument displays. The out-the-window display and simulation models are driven by a Silicon Graphics Onyx® with a RealityEngine™, 2 raster managers, and 2 MIPS® R8000™ 75-MHz processors with 64 Mbytes RAM.[a] The instrument displays are driven by a Silicon Graphics Indigo2™ workstation. A user interface is provided using a Silicon Graphics Indy™ workstation. The instrument displays and the user interface are programmed in the VAPS™ GUI toolset, from Virtual Prototypes Inc.[b]

Design Requirements

The requirements for the code processing and data processing software suite developed to support the REFS may be summarized as follows:

- The software must provide a means of integrating the simulation models produced by the ISI toolset with the software already installed on the REFS.
- The software must be capable of generating simulation initial conditions from the trim flight configuration specified by the user and be capable of initializing the simulation at the specified trim condition. The term trim is used here in a general sense to mean initializing the simulation at the steady state solution resulting from any specified flight condition.
- A user interface must be provided to setup the simulation and launch it. This setup includes the specification of the trim flight condition. The user interface has to be exceptionally easy to use, simple and customizable.
- output from the simulation must be usable by the design software for analysis.

Integration of the REFS Simulator with the Design Environment

The software integration process has a number of well-defined component steps. These are illustrated in Figure 2. Each step (except the development phase) shown in Figure 2 required the in-house development of supporting software utilities that, in sum, comprise the software suite that processes code and data. An explanation of each step, and how it addresses the design requirements, follows.

Development Phase

The integration process begins with the proper preparation of a model in SystemBuild™ on an engineer's desktop computer. Seamless integration of the simulation model with the other software

[a] Reality Engine, Indigo2 and Indy are trademarks of Silicon Graphics Inc. Onyx is a registered trademark of Silicon Graphics Inc.

R8000 is a trademark of MIPS Technologies Inc. MIPS is a registered trademark of MIPS Technologies Inc.

[b] VAPS is a trademark of Virtual Prototypes Inc.

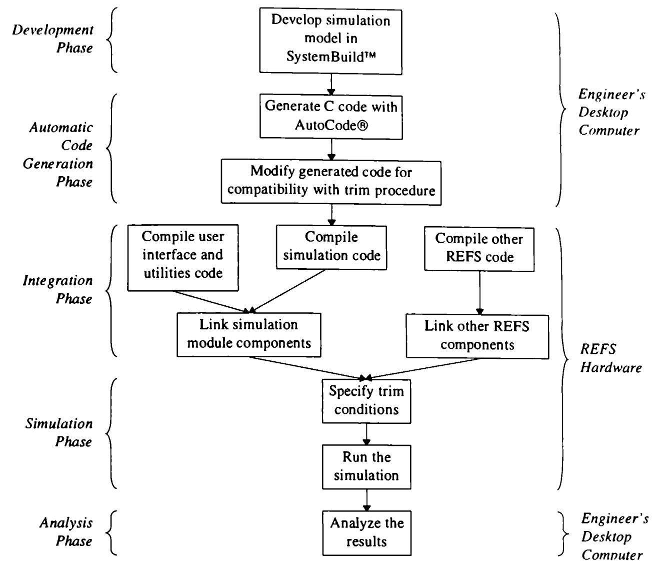

Development Phase
Automatic Code Generation Phase
Integration Phase
Simulation Phase
Analysis Phase
Develop simulation model in SystemBuild™
Generate C code with AutoCode®
Modify generated code for compatibility with trim procedure
Compile user interface and utilities code
Compile simulation code
Compile other REFS code
Link simulation module components
Link other REFS components
Specify trim conditions
Run the simulation
Analyze the results
Engineer's Desktop Computer
REFS Hardware
Engineer's Desktop Computer

Figure 2, Integration of the REFS Simulator with the Design Environment

components of the simulator requires that information on a large number of model parameters must be passed from the model to the other simulation programs; and that the model, when converted to C code, may be linked together with the other simulation programs.

In order to ensure proper linking of the model with other software components of the simulation, a standard naming convention needs to be adopted for model inputs, states and outputs so that information is properly passed to other subroutines. To allow for the monitoring of a large number of output variables without having to endure the inconvenience of connecting output signals many levels down in the model hierarchy to external outputs, Write to Variable (WTV) blocks were used. These are translated into global variables in the generated C code, which makes them easy to access. For the sake of consistency, WTV variables are considered to be the only outputs of the simulation model and are the only outputs used by the other software modules.

The need for producing code that can be run in real-time also affects the way in which the simulation model is structured in SystemBuild™. Discrete-time subsystems tend to translate into large C functions after using AutoCode®, however, discrete-time procedures do not. The size of the resultant C functions may therefore be kept small by using Procedure superblocks in the model. This results in the production of a larger number of smaller functions, which are easier to optimize by the C compiler for increased execution speed.

Automatic Code Generation Phase

The AutoCode® utility provides the capability to generate a C code representation of the simulation model. However, this code is not fully compatible with the REFS software when the default ISI code generation templates are used. In order to meet the requirement of integration with the REFS simulator, it is necessary to design custom templates and define a new code generation method, which proceeds in two steps.

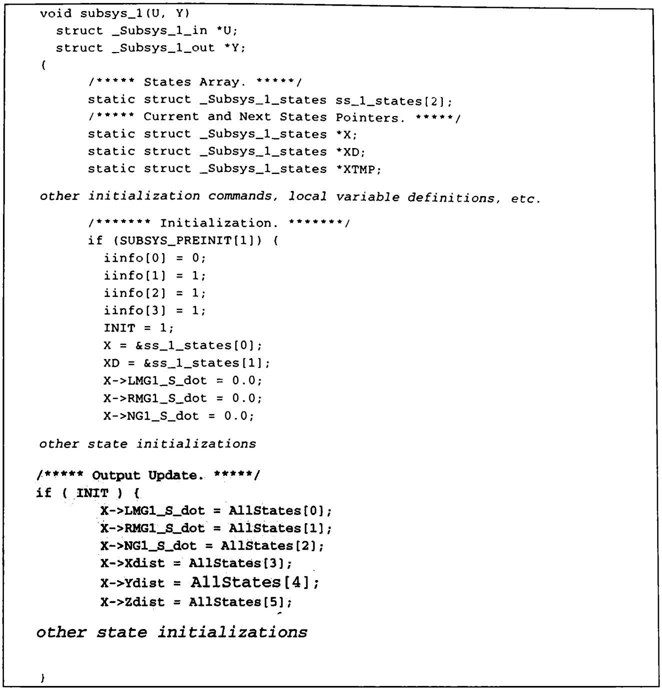
In the first step, AutoCode® is used together with the in-house templates to generate an initial version of the simulation code as well as a suite of C programs, which are used in the second step to further modify the generated code. The result is a set of source files that are suitable for linking with the other components of the REFS simulator. The second step is necessary because the variables representing the model states appear as local variables within the subsystem functions in the generated C code for discrete-time models. They can therefore not be accessed from outside of those functions. This means that the initial values of the model states are fixed to some constants in the generated code, which violates the required capability to initialize the simulation in any flyable condition using an external trim algorithm. Figure 3'' shows an example of a subsystem function generated by the AutoCode® utility and Figure 4 shows how this code is adapted to provide access to the state variables from outside the subystem function through the use of a global array of initial state values called *AllStates*. The shaded area shows the code that is added during the postprocessing phase.

In order that the simulation code may interface with an external trim program, it is necessary to add the calculation of state derivatives in the subsystem functions. The code processing programs add a global array variable called *AllStateDerivatives* and insert code automatically in each subsystem function. This code initializes the array with the values of the state derivatives. Since discrete-time models are used, the derivatives are calculated using the following approximation:

$$\dot{x}_k = \frac{x_{k+1} - x_k}{\Delta T} \qquad (1)$$

```
void subsys_1(U, Y)
  struct _Subsys_1_in *U;
  struct _Subsys_1_out *Y;
{
      /***** States Array. *****/
      static struct _Subsys_1_states ss_1_states[2];
      /***** Current and Next States Pointers. *****/
      static struct _Subsys_1_states *X;
      static struct _Subsys_1_states *XD;
      static struct _Subsys_1_states *XTMP;

other initialization commands, local variable definitions, etc.

      /******* Initialization. *******/
      if (SUBSYS_PREINIT[1]) {
        iinfo[0] = 0;
        iinfo[1] = 1;
        iinfo[2] = 1;
        iinfo[3] = 1;
        INIT = 1;
        X = &ss_1_states[0];
        XD = &ss_1_states[1];
        X->LMG1_S_dot = 0.0;
        X->RMG1_S_dot = 0.0;
        X->NG1_S_dot = 0.0;

other state initializations

mathematical model
}
```

Figure 3, Example of Subsystem Code Generated by AutoCode®



```
void subsys_1(U, Y)
  struct _Subsys_1_in *U;
  struct _Subsys_1_out *Y;
{
      /***** States Array. *****/
      static struct _Subsys_1_states ss_1_states[2];
      /***** Current and Next States Pointers. *****/
      static struct _Subsys_1_states *X;
      static struct _Subsys_1_states *XD;
      static struct _Subsys_1_states *XTMP;

other initialization commands, local variable definitions, etc.

      /******* Initialization. *******/
      if (SUBSYS_PREINIT[1]) {
        iinfo[0] = 0;
        iinfo[1] = 1;
        iinfo[2] = 1;
        iinfo[3] = 1;
        INIT = 1;
        X = &ss_1_states[0];
        XD = &ss_1_states[1];
        X->LMG1_S_dot = 0.0;
        X->RMG1_S_dot = 0.0;
        X->NG1_S_dot = 0.0;

other state initializations

/***** Output Update. *****/
if ( INIT ) {
        X->LMG1_S_dot = AllStates[0];
        X->RMG1_S_dot = AllStates[1];
        X->NG1_S_dot = AllStates[2];
        X->Xdist = AllStates[3];
        X->Ydist = AllStates[4];
        X->Zdist = AllStates[5];

other state initializations

}
```

Figure 4, Example of Subsystem Code After the Postprocessing Phase

Interfacing with the trim algorithm also requires that the inputs and state values be passed by the trimming algorithm to the simulation model, and that the state derivatives and outputs be passed back to the trimming algorithm. Due to the multi-rate nature of the simulation model, AutoCode® partitions it into multiple subsystem procedures with buffering between them. This means that feedback from one subsystem to another must be accomplished by copying the data from the output buffer of one subsystem to the input buffer of the next. In order to obtain the final value of all outputs and state derivatives, it is necessary to repeatedly copy the buffers and execute the subsystem functions until the state derivatives and outputs reach stable values. This method works as long as there is no algebraic loop in the model and that all random processes are turned off during the trim, which is not a problem in practice.

The custom templates also produce a look-up table of (*variable name*, *pointer*) pairs which are used to implement run-time indirect addressing of the simulation model variables. This is necessary since the indices of the state variables have a tendency to vary following modifications made to the simulation model in SystemBuild™. Run-time linking solves this problem and allows for the simulator code to be fairly resistant to changes made in SystemBuild™. Another benefit of this approach is the added capability to add new inputs, states and outputs to the model without requiring modifications to other components of the simulator. The user interface is developed in a way to automatically recognize these variables and allow the user to access them by their name.

Integration Phase

In this phase, the modified simulation code is compiled and linked with programs that implement a user interface and a trim algorithm. The resulting software module can interface with the other components of the REFS simulator, thus satisfying the first design requirement.

In order to satisfy the second requirement, a trim program is needed. This program receives a flight condition specification from the user and then initializes the simulation to that condition. Once the trim is complete, normal flight may proceed. The trim algorithm used is given in Figure 5 below.

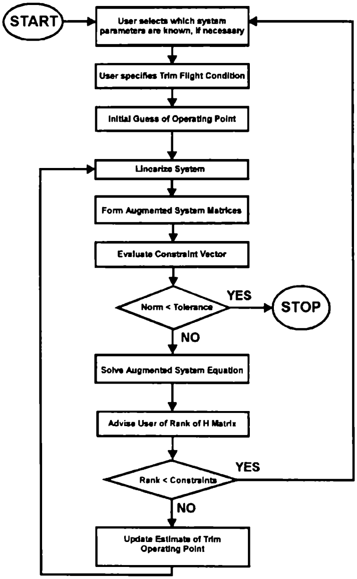

START
User selects which system parameters are known, if necessary
User specifies Trim Flight Condition
Initial Guess of Operating Point
Linearize System
Form Augmented System Matrices
Evaluate Constraint Vector
Norm < Tolerance
YES
STOP
NO
Solve Augmented System Equation
Advise User of Rank of H Matrix
Rank < Constraints
YES
NO
Update Estimate of Trim Operating Point

Figure 5, Schematic of Trimming Algorithm

The trim configuration is specified in two steps. In step one, a determination is made of which inputs and states are known and which have to be calculated; and which state derivatives and outputs are known, and which are inconsequential and may be ignored. The trim program performs much of this determination of active and in-active inputs, states, derivatives and outputs through the use of pre-programmed algorithms. However, these default algorithms are based on particular assumptions of the initial flight condition that is required. To make this procedure more general, the user may override the defaults and make alternate assignments.

In step two, the user needs only to specify the aircraft airspeed, altitude, weight, centre-of-gravity position, flap and landing gear settings, flight path angle, sideslip angle and basic configuration information (such as whether yaw dampers are on or off). The simulation program reads the input file and then assigns numerical values to those inputs, states, state derivatives and outputs which are known apriori, using pre-programmed algorithms. Again, to make the procedure more general, the user is provided with the capability of overriding the default set of constraints when necessary through the use of an input file.

The trim methodology is similar to that used by the ISI design tools. The trim equations are formulated as a Newton-Raphson root solving problem.[1] This is accomplished by linearizing the system about the initial guess of the trim solution.[2] Assume that the non-linear system of interest may be represented by the following equations:

$$\dot{x} = f(x,u) \tag{2}$$

$$y = g(x,u) \tag{3}$$

where x are the system states, u are the system inputs, and y are the system outputs. The right hand sides of equations (2) and (3) may be approximated with Taylor series:

$$\dot{x} = f(x_0,u_0) + \frac{\partial f(x_0,u_0)}{\partial x}\Delta x + \frac{\partial f(x_0,u_0)}{\partial u}\Delta u + \ldots \tag{4}$$

$$y = g(x_0,u_0) + \frac{\partial g(x_0,u_0)}{\partial x}\Delta x + \frac{\partial g(x_0,u_0)}{\partial u}\Delta u + \ldots \tag{5}$$

where

$$x = x_0 + \Delta x \tag{6}$$

$$u = u_0 + \Delta u \tag{7}$$

and x_0 , u_0 represent the operating point about which the system is to be linearized. By defining

$$A = \frac{\partial f(x_0,u_0)}{\partial x}, \quad B = \frac{\partial f(x_0,u_0)}{\partial u},$$

$$C = \frac{\partial g(x_0, u_0)}{\partial x} \text{ , and } D = \frac{\partial g(x_0, u_0)}{\partial u} \qquad (8)$$

and ignoring higher order terms, equations (4) and (5) may be recast into the following form:

$$A\Delta x + B\Delta u = \dot{x} - f(x_0, u_0) \qquad (9)$$

$$C\Delta x + D\Delta u = y - g(x_0, u_0) \qquad (10)$$

Equations (9) and (10) are then combined into augmented matrices to form the following equation:

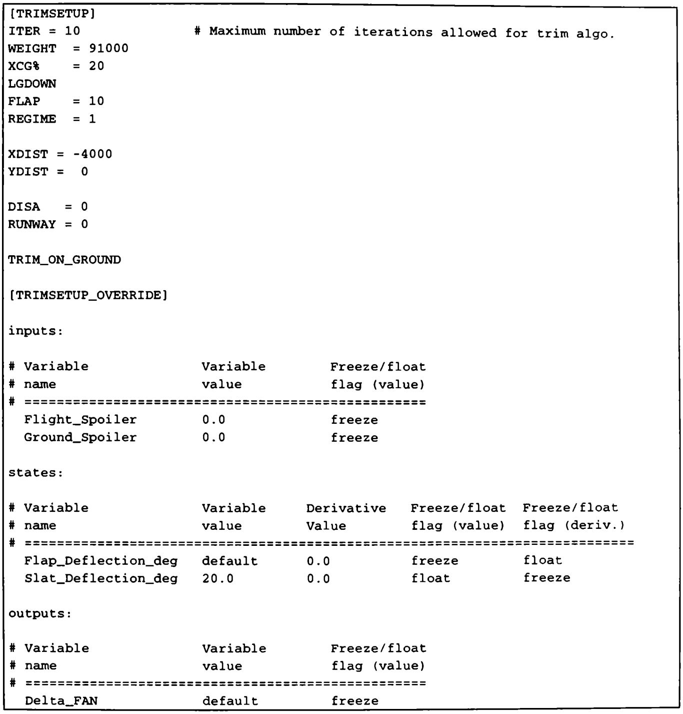
$$H \begin{bmatrix} \Delta x \\ \Delta u \end{bmatrix} = G \qquad (11)$$

where

$$H = \begin{bmatrix} A & B \\ C & D \end{bmatrix} \qquad (12)$$

$$G = \begin{bmatrix} \dot{x} - f(x_0, u_0) \\ y - g(x_0, u_0) \end{bmatrix} \qquad (13)$$

Row and column eliminations are performed to remove the inputs and states on the left-hand side of equation (11) for which trimmed values are known; and to remove the unknown state derivatives and outputs on the right-hand side of equation (11) that are inconsequential to the solution of the equation. The row vector, G, is called the "constraint vector". The values of $\dot{x}$ and y in the reduced equations are



```
[TRIMSETUP]
ITER = 10               # Maximum number of iterations allowed for trim algo.
WEIGHT  = 91000
XCG%    = 20
LGDOWN
FLAP    = 10
REGIME  = 1

XDIST = -4000
YDIST =  0

DISA   = 0
RUNWAY = 0

TRIM_ON_GROUND

[TRIMSETUP_OVERRIDE]

inputs:

# Variable                Variable        Freeze/float
# name                    value           flag (value)
# ================================================
  Flight_Spoiler          0.0             freeze
  Ground_Spoiler          0.0             freeze

states:

# Variable                Variable     Derivative    Freeze/float  Freeze/float
# name                    value        Value         flag (value)  flag (deriv.)
# ===========================================================================
  Flap_Deflection_deg     default      0.0           freeze        float
  Slat_Deflection_deg     20.0         0.0           float         freeze

outputs:

# Variable                Variable        Freeze/float
# name                    value           flag (value)
# ================================================
  Delta_FAN               default         freeze
```

Figure 6, Example of Simulator Input File

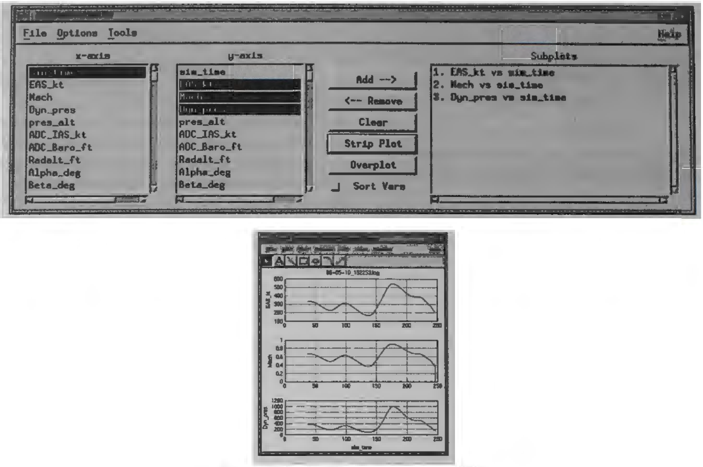


Figure 7, Simplot for Xmath™ Utility

known — these are the desired rates and outputs of the trimmed system. Thus, as the choice of operating point approaches the actual trim states and inputs,

$$G \rightarrow 0 \quad \text{as} \qquad (x_0, u_0) \rightarrow (x_{trim}, u_{trim}) \qquad (14)$$

The constraint vector is thus a measure of the convergence of the Newton-Raphson root finding algorithm. The Euclidean norm of the constraint vector is evaluated. If this norm is less than a hard-coded tolerance, then the algorithm is said to have converged and execution is discontinued.

Otherwise, equation (11) is solved for Δx and Δu using a singular value decomposition algorithm.[3] The rank of matrix H is also checked by the singular value decomposition algorithm, which provides a check on whether or not a solution exists. If the rank of H is less than the number of constraints (that is, the number of elements of the constraint vector), then equation (11) is over-constrained, or the wrong inputs, states, state derivatives or outputs were removed during the model order reduction step. The user is informed of the rank of the matrix H, the number of constraints and the number of unknowns (the number of elements of the vector $[\Delta x \quad \Delta u]^T$). However, the trim algorithm will proceed if the rank of H is less than the number of constraints since an approximate solution may usually be found. At this point, the user has the choice of terminating the trim algorithm and correcting the input files, or allowing the algorithm to proceed.

The solution of equation (11) provides values for Δx and Δu which are substituted into equations (6) and (7) to provide a value for the state and input vectors x and u, respectively. The new values for states and inputs are then used for another iteration of the trim algorithm as the new operating point. The process described above is repeated until the trim operating point is found.

Simulation Phase

The third requirement is that a user interface be developed in order to provide a simple and user-friendly means for someone to specify the trim conditions and initialize the simulator. For the sake of simplicity and flexibility, a text file-oriented approach was taken. Figure 6 shows a typical input file which specifies the trim conditions for a test flight. Figure 6 also provides an example of how a simulator user may override the default trim setup algorithms contained in the simulator software. Spoiler, flap, slat and engine fan settings are set in this example by the user. Normally, default values would be assigned to these parameters.

Analysis Phase

During the simulation, the data recording component of the REFS software writes to a file the values of various parameters of interest. The last design requirement is that a way of bringing these results back into the Xmath™ environment be provided.

To achieve this goal, a utility was written to be run in Xmath™ that can read data produced by the REFS. A simple GUI interface allows the user to choose which parameters are to be plotted and how (either versus time, or versus any other parameter) (see Figure 7). Multiple parameters may be displayed on one plot. The tool may also be used to average the value of a parameter over a particular time period and plot this versus any other parameter (which is also averaged, automatically, over the same time period). The selection of successive time intervals causes successive points to be added to the cross plot. For example, plots of stick force versus speed can be quickly constructed directly from simulator data using this capability. With Xmath™ running concurrently with the simulation, plots may be produced during a test session. Since Xmath™ is also used in the design departments, it is also possible for the design engineers to have immediate access to data generated by the simulator. The data need only be copied over the LAN network to the design engineer's computer and then analyzed.

Numerical output and plots, updated in real-time, may also be programmed on the REFS' operators console. The only limitation is that the selection of parameters being displayed may only be changed by recoding and recompiling the GUI software for the console.

Conclusion

A software suite was developed that meets all of the design requirements set forth in this paper. This software has been used operationally on the REFS simulator. The software suite, in combination with the ISI tools and standard REFS software, produces an operating environment that is fully integrated; allows the user to setup the simulation and modify trim conditions; initializes the simulation at specified trim points; and permits design engineers to use the simulation output data directly. This software suite has made it possible to use models developed with SystemBuild™ on a fixed-base flight simulator.

Furthermore, the process of installing a simulation model developed with SystemBuild™ into the REFS simulator is very quick and simple, requiring very little user input. Consequently, time savings have been realized in the overall process of updating a simulation model and installing it on the REFS simulator. Performing these simulations and gaining access to their results is equally quick and efficient.

References

[1] W.H.Press, B.P.Flannery, S.A.Teukolsky, W.T.Vetterling, *Numerical Recipes in C, The Art of Scientific Computing*, Cambridge University Press, 1988, pg.286

[2] Integrated Systems Inc., *SystemBuild Reference Guide*, Santa Clara, California, USA, 1995, pg.71

[3] W.H.Press, B.P.Flannery, S.A.Teukolsky, W.T.Vetterling, Numerical *Recipes in C, The Art of Scientific Computing*, Cambridge University Press, 1988, pg.60

AIAA-96-3513-CP

MULTIPROCESSOR COMPUTATION IN THE RESEARCH FLIGHT SIMULATION ENVIRONMENT

Matthew W. Blake, NASA Ames Research Center
Joseph R. King, NSI Technology Services Corporation
Craig Pires, NSI Technology Services Corporation

ABSTRACT

This paper describes a new multiprocessor implementation of a very complex research flight simulation using essentially Commercial Off The Shelf (COTS) hardware and system software. A description of the simulator and it's primary components is provided followed by a description of the real-time system as implemented. The real-time system spans three synchronized processors and can be expanded to 11 processors on the current hardware platform. The implementation has provided a common environment between the multi-processor real-time computer, integration and test station, and desktop development systems. The system has provided significant performance improvements and increased efficiency of operations and support. Several potential improvements are also described.

INTRODUCTION

In the aircraft training simulator market, simulations often run on multi-processor computers to gain the cost efficiency of fewer computer systems utilizing less expensive processors. The simulation developer incurs the time and expense of partitioning the software and verifying the complex interaction and timing between processors only once, then numerous identical copies of the training simulator are sold. In the research environment the software is constantly being modified to perform a new function. Under this environment, it has traditionally been easier to develop and test the simulation by having the code run sequentially on one processor. This avoided the difficult verification task of ensuring each process was independently functioning correctly and that each process was communicating in the proper sequence with the other processes. Unlike the training market, in the research environment there is no economy of scale for any extra expense in developing and testing the simulation across multiple processors since additional copies of the simulation are not created and sold.

When used for research purposes, flight simulation has generally focused on one flight regime and only that part of the vehicle and it's systems are simulated. In the case of full-mission simulation of a commercial transport, the entire environment that the crew experiences must be provided. If all systems are simulated (no actual avionics boxes), the size of the simulation software gets extremely large. The Advanced Concepts Flight Simulator (ACFS) is one such simulation, representing a complete full-mission simulation of a commercial transport aircraft and it's related systems and environment. The ACFS software includes approximately 500,000 lines of code, split approximately evenly between the programming languages FORTRAN and C. The ACFS is one of two full-mission flight simulators at the Crew-Vehicle Systems Research Facility (CVSRF) at NASA Ames Research Center. The ACFS is used to study aspects of human factors in aviation safety as well as methods to improve aviation operational efficiency.

The ACFS simulates a generic advanced twin engine mid-range transport (similar to a Boeing 757). The ACFS consists of a reconfigurable cab on a 6 degree-of-freedom motion platform with a color night/dusk Out-The-Window (OTW) visual system. The cockpit includes control sticks and rudder pedals, conventional center console with throttle levers and control panels, an overhead panel for control of other aircraft subsystems, and 8 video display screens across the main panel for flight, guidance, navigation, and status displays. There is a Mode Control Panel (MCP) on the glare-shield for autoflight control and two Control Display Units (CDU) for interfacing with the Flight Management Computer (FMC) simulation. This configuration provides full-mission aircraft functionality including complete auto-flight, auto-throttle and Flight Management System (FMS) capability, yet the hardware and software configuration can be changed as needed to address specific research requirements. The ACFS can be configured to operate with the other CVSRF simulator, a Federal Aviation Administration (FAA) certified Boeing 747-400 simulator as well as an elaborate Air Traffic Control (ATC) simulator. Unlike most training simulators and the 747-400 simulator at the CVSRF, the ACFS does not include any actual aircraft avionics boxes so the entire simulation is fully programmable to address any research requirements. Figure 1 shows a system diagram of the main components of the ACFS.

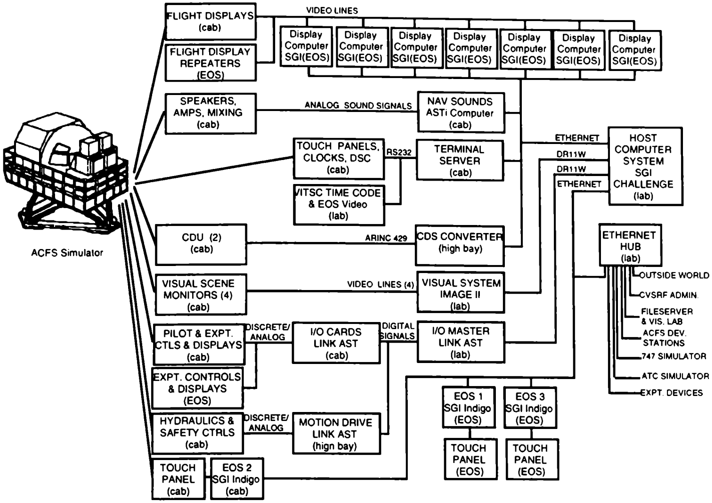

FLIGHT DISPLAYS (cab)
FLIGHT DISPLAY REPEATERS (EOS)
VIDEO LINES
Display Computer SGI(EOS)
Display Computer SGI(EOS)
Display Computer SGI(EOS)
Display Computer SGI(EOS)
Display Computer SGI(EOS)
Display Computer SGI(EOS)
Display Computer SGI(EOS)
SPEAKERS, AMPS, MIXING (cab)
ANALOG SOUND SIGNALS
NAV SOUNDS ASTi Computer (cab)
TOUCH PANELS, CLOCKS, DSC (cab)
RS232
TERMINAL SERVER (cab)
ETHERNET
DR11W
DR11W
ETHERNET
HOST COMPUTER SYSTEM SGI CHALLENGE (lab)
VITSC TIME CODE & EOS Video (lab)
ACFS Simulator
CDU (2) (cab)
ARINC 429
CDS CONVERTER (high bay)
ETHERNET HUB (lab)
VISUAL SCENE MONITORS (4) (cab)
VIDEO LINES (4)
VISUAL SYSTEM IMAGE II (lab)
OUTSIDE WORLD
CVSRF ADMIN.
FILESERVER & VIS. LAB
ACFS DEV. STATIONS
747 SIMULATOR
ATC SIMULATOR
EXPT. DEVICES
PILOT & EXPT. CTLS & DISPLAYS (cab)
DISCRETE/ ANALOG
I/O CARDS LINK AST (cab)
DIGITAL SIGNALS
I/O MASTER LINK AST (lab)
EXPT. CONTROLS & DISPLAYS (EOS)
EOS 1 SGI Indigo (EOS)
EOS 3 SGI Indigo (EOS)
HYDRAULICS & SAFETY CTRLS (cab)
DISCRETE/ ANALOG
MOTION DRIVE LINK AST (hign bay)
TOUCH PANEL (EOS)
TOUCH PANEL (EOS)
TOUCH PANEL (cab)
EOS 2 SGI Indigo (cab)

Figure 1: ACFS System Diagram

The ACFS was originally built in the mid-1980s and has gone through many upgrades and enhancements since. This paper addresses the recent replacement of a late 1980s Digital Equipment Corp. (DEC) VAX computer. In replacing this computer system, the goal was to increase reliability, provide significant spare processor and memory capacity, and greatly increase the efficiency in developing, testing, and operating simulation experiments without creating extensive custom real-time control software. The goals of increased reliability and direct performance (processor speed and memory) were easy to achieve as many computers provide improvements in these areas over the mid-1980s VAX that was being replaced. To achieve the goal of improved productivity using COTS systems in a research flight simulation environment led to the purchase of a Silicon Graphics Inc. (SGI) Challenge host computer and a set of SGI workstations.

HOST COMPUTER ARCHITECTURE

The real-time host computer is an SGI Challenge L multi-processor computer capable of housing 12 processors. The current implementation described here utilizes only four of the available processor slots. Each of the processors is a model R4400 with a 200 MHz. clock speed. The supporting single processor SGI workstations include Indy, Indigo, and Indigo2 models with various processors.

MEMORY MANAGEMENT

The SGI Challenge host computer is configured with 128 Megabytes of main memory, expandable to 2 Gigabytes. All critical program variables and constants are assembled in one section of shared memory called Global Common. The current Global Common utilizes approximately 634K bytes of memory. The Global Common memory is partitioned into sections along aircraft systems boundaries such as navigation systems, flight dynamics, etc. A common Global Common file is processed into both C and FORTRAN INCLUDE structure files so that the C and FORTRAN routines will map to the same variables. Other processes, executing in other processors, that access the shared memory include the data collection write to disk process, the Experimenter/Operator Station (EOS) processes, and the FMC process.

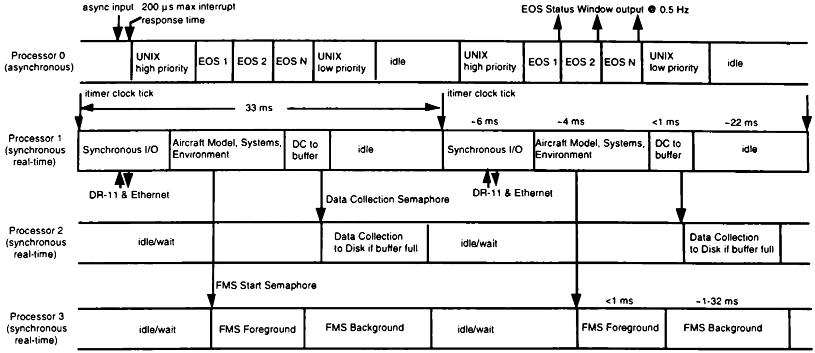


Figure 2: Sample processor activity time-line

PROCESS CONTROL AND SEQUENCING

SGI's IRIX (Version 5.3) Operating System (OS) offers a number of services and routines to control the execution of processes running on their multi-processor computers. These services include:

- Locking processes into memory
- Isolating processors from any external activity and allowing only the selected processes to run on those processors
- Setting process priorities that will run at levels higher than any other "User" process and not be adjusted
- Synchronization of processes on different processors

Using these OS features provides the ability to force real-time behavior on all processors other than the one running non-real-time UNIX processes. The processor load is currently split as follows:

- Processor 0: UNIX OS, EOS, Asynchronous Input/Output (I/O)
- Processor 1: Synchronous I/O, Main simulation calculations
- Processor 2: Data Collection
- Processor 3: FMC process

Processors 1 - 3 are run in an isolated mode with only the specified processes locked in place and at the highest priority. Since some important simulator functions are still performed on processor 0, a "Single User" mode was developed. In this mode, all non-essential processes and services are disconnected from the simulator. Outside network activity is minimized by isolating the real-time networks with filtering bridges and routers within the Ethernet communication hub. At simulation start-up, all essential files are copied to local disks so that simulator operations can be performed independent of the shared file-server system by unmounting the file-server disks from the real-time host. This isolation mode allows full use of the file-server for development while minimizing impact to the real-time simulation.

Frame timing is currently provided by the ITIMER utility providing a resolution approaching 1 microsecond.

Communication between the various real-time processes running on different processors is via shared memory Global Common (GC) with synchronization using SGI Arena/Semaphore signals. The Arena/Semaphore signals have a guaranteed latency of less than 200 microseconds and in practice have provided approximately 10 microseconds latency. Figure 2 shows a high level diagram of the time line of two real-time frames with the process semaphore signals and main processor activities.

MAIN MODEL & SYNCHRONOUS I/O PROCESS

The main aircraft model and synchronous I/O functions are performed by processor 1. The model is partitioned into approximately 30 subsystems such as aerodynamics, engines, FMC communication, I/O to instruments, etc. Each subsystem may contain hundreds of subroutines. Execution of these subsystems is controlled by the model scheduler (MODSCH). MODSCH controls in what order and how often each subsystem is executed.

MODSCH determines this schedule based on an ASCII database called the Model Attribute Table (MAT). By modifying this MAT database, operators can quickly change the characteristics of the simulator by enabling/disabling modules or increasing/decreasing the execution rate. The 30 subsystems are partitioned into approximately 120 MAT table entries to increase scheduling flexibility. The highest frequency loop is currently 30 Hz (33 ms per frame). Figure 3 shows a high level diagram describing one 33 ms frame of MODSCH. Figure 4 shows the sequencing attributes of one subsystem as described in the MAT table as well as the definitions of MAT table options.

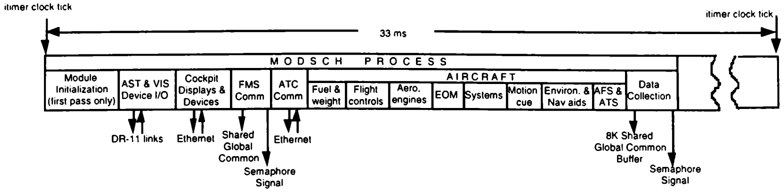


Figure 3: MODSCH timeline for one frame

MAT.DAT TABLE EXTRACT:

```
        •
        •
        •
        •
        •
        •
        •

MODEL_NAME:               FLAPSLAT
ENABLED                   ENABLED
FRAMES:                   MASK 010000010000010000010000010000
GROUP:                    FCSURF
TIMES_CALLED_INIT         2
DELAY_BEFORE_RUN:         NO
RUN_DURING_FREEZE         NO
RUN_DURING_RESET          YES
RUN_AFTER_EXCEPTION       NO
RUN_DURING_STOP           NO

MODEL_NAME:               SPOILERS
ENABLED:                  ENABLED
FRAMES                    MASK 000010000010000010000010000010
GROUP:                    FCSURF
TIMES_CALLED_INIT:        2
DELAY_BEFORE_RUN:         NO
RUN_DURING_FREEZE         NO
RUN_DURING_RESET          YES
RUN_AFTER_EXCEPTION       NO
RUN_DURING_STOP:          NO

MODEL_NAME                NEW_AEROCS
ENABLED:                  ENABLED
FRAMES:                   ALL
GROUP:                    AERO
TIMES_CALLED_INIT         10
DELAY_BEFORE_RUN:         NO
RUN_DURING_FREEZE         NO
RUN_DURING_RESET:         YES
RUN_AFTER_EXCEPTION       NO
RUN_DURING_STOP           NO

        •
        •
        •
        •
        •
        •
        •
```

MAT.DAT TABLE DEFINITIONS:

MODEL_NAME - Function name as listed in MODEL.STC (max. 15 bytes)

ENABLED - Model is active this run: ENABLED - the model will be called as requested.
DISABLED - in the MAT but disabled (i.e. not called).

FRAMES - Which frames model is called; held in a 30 element mask.
ALL - all frames (111111 ...)
EVEN - all even frames (i.e. 101010 . .)
ODD - all odd frames (i.e. 010101 ...)
NONE - only called during initialization mode.
MASK - a binary mask (e.g. MASK 001001001001)
SEQ - a number sequence (e.g. SEQ 1,5,8)

GROUP - the name of a model group defined in file acfs_dat: MODEL_GROUP_FILE.DAT. Each model is associated with one group. The default is NONE. Models should be assigned to groups of related models.

TIMES_CALLED_INIT - How many times model should be called during initialization mode. (Range: 0 through 200)

DELAY_BEFORE_RUN - a model can be temporarily kept from running immediately after initialization or after each reset for a preset number of cycles. Each process can be set to delay a different number of cycles.

YES - delay this model before running
NO - run this model according to schedule

RUN_DURING_FREEZE - model continues to be scheduled during freeze mode.

YES - call this routine during freeze mode
NO - Don't call this routine during freeze mode.

RUN_DURING_RESET - model is called with initialization mask during RESET mode.

YES - call this routine during RESET mode.
NO - Don't call this routine during RESET mode.

RUN_AFTER_EXCEPTION - model continues to be scheduled even after an exception has been caught (e.g. zero divide).
YES - continue calling this routine despite error.
NO - stop scheduling this model after an exception.

RUN_DURING_STOP - model is called once while the simulation is stopping
YES - call this routine during STOP mode.
NO - Don't call this routine during STOP mode

Figure 4: Sample MAT table and MAT table definitions

FMC PROCESS

The FMC simulation is computed on processor 3. The FMC simulation is run as two processes; a Foreground process and a Background process. The Foreground process handles the I/O to the CDUs, I/O to the aircraft model, and computation of guidance information (comparison of actual to planned path and needed corrections). The Foreground process is run synchronously with the aircraft model running in processor 1. This provides real-time response to crew actions (keystrokes) and for continuous guidance to the aircraft autoflight system. Communication of data is through Global Common shared memory and initiation of the FMC Foreground process is through semaphore signals. The Background process handles flight planning which includes establishing way point and path information, aircraft performance computations, navigation data processing, and processing the CDU page formatting. The Background process runs in an asynchronous mode, and runs whenever calculations are required and when the Foreground process is not running. This design is consistent with the software operation of most commercial transport aircraft FMC avionics systems.

DATA COLLECTION PROCESS

To conduct human factors research, large amounts of information regarding complete status of the simulated aircraft systems and all crew activities must be recorded for post simulation analysis. To accomplish this requirement the ACFS Challenge host computer system was configured with sufficient hard disk capacity to save the collected experiment data during the real-time simulator operation. The current configuration provides 2 Gigabytes of space for data collection. Software tools have been developed to post-process the data into other formats to meet researcher needs or provide a "quick-look" at the data for verification.

Implementation of the data collection software was accomplished using three run-time modules and an off-line data definition compiler. The host initialization module (DRG_INIT) and the real-time module (DRH_REC) runs in processor 1 within the main real-time load. A slave process that performs disk I/O (DRG_SLAVE) runs on processor 2.

Setup of data collection begins with a specification of the desired simulation variables and collection rate (between 1 Hz and 30 Hz). This list is edited into an ASCII data definition file with the suffix of .rec. This file is compiled with the off-line data definition compiler (DRC) program to create the runtime files with the .dra, .drs, and .log suffixes.

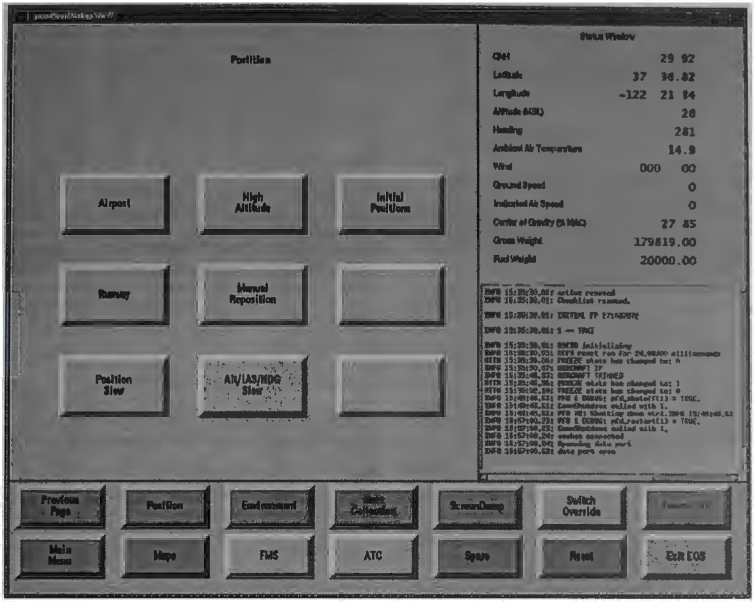
During real-time operation an 8K byte buffer residing in the Global Common shared memory area is packed with the predefined experiment data at the end of each 30 Hz frame by the DRH_REC module running in processor 1. The DRH_REC module then sends a SGI Arena/Semaphore signal to the DRG_SLAVE module running on processor 2. The DRG_SLAVE module then starts up and moves the 8K byte buffer into one of two 121K byte disk I/O buffers and checks to see if the 121K buffer is full. If it is full, DRG_SLAVE identifies the alternate I/O buffer as the current collection buffer for use at the end of the next simulation frame. DRG_SLAVE then initiates a disk write operation of the full buffer. This utilization of two buffers is called double buffering.

Run-time control of the data collection system is through the EOS. The data collection file to be written to disk is comprised of three main sections. The first section contains header information which is entered from the EOS and includes the simulator crew names and/or numbers, experiment number, run number, date, etc. The second section contains definitions of the data to be collected which is taken directly from the pre-processed list. The third and largest section of the data file is the actual experiment run data collected in real-time.

EXPERIMENTER/OPERATOR STATION PROCESSES

The Experimenter/Operator Station (EOS) currently runs on processor 0. The process runs at a high priority non-real-time status. The EOS is similar to the Instructor/Operator Station (IOS) on a conventional training flight simulator but features have been added to the ACFS EOS to support the research requirements. The ACFS EOS configuration was designed to allow complete simulator control from the on-board EOS station, at the rear area of the cockpit enclosure, or from a remote control room area located within the facility. Any number of EOS processes can run concurrently so several users can monitor different simulator activity simultaneously.

SGI Indy workstations were selected for the EOS to provide the simulator user the desired Graphical User Interface (GUI). The EOS process actually runs in the SGI Challenge host computer via remote login. Communication between the EOS Indy systems and the host uses standard Terminal Control Protocol/Internet Protocol (TCP/IP) protocols imbedded within the SGI computer OS. The host resident EOS processes are executed at a high priority non-real-time mode. Touch sensitive screens were installed on each of the 19" EOS displays and the display format was designed to use the touch screen or trackball/mouse input to activate functions and enter data.



```
Status Window
QNH                                 29 92
Latitude                        37  36.82
Longitude                     -122  21 54
Altitude (MSL)                         26
Heading                               281
Ambient Air Temperature              14.9
Wind                             000   00
Ground Speed                            0
Indicated Air Speed                     0
Center of Gravity (% MAC)           27 85
Gross Weight                    179819.00
Fuel Weight                      20000.00
```

```
INFO 15:35:30.01: action removed
INFO 15:35:30.01: Checklist removed.

INFO 15:35:30.01: INITIAL FP [illegible]

INFO 15:35:30.01: 1 -- [illegible]

INFO 15:35:30.01: [illegible] initializing
INFO 15:35:30.03: [illegible] reset [illegible]
WTTR 15:35:30.06: FREEZE state has changed to: 0
INFO 15:35:50.07: AIRCRAFT IF
INFO 15:35:48.83: AIRCRAFT TRIMMED
WTTR 15:35:46.36: FREEZE state has changed to: 1
WTTR 15:35:50.10: FREEZE state has changed to: 0
INFO 15:48:46.61: [illegible] = TRUE.
INFO 15:48:48.51: [illegible] called with 1.
INFO 15:48:45.61: [illegible] Shutting down [illegible] INFO 15:48:45.52
INFO 15:57:40.21: [illegible] restart[1] = TRUE.
INFO 15:57:40.23: [illegible] called with 1.
INFO 15:57:40.24: socket connected
INFO 15:57:40.54: Spawning data port
INFO 15:57:40.62: data port open
```

Figure 5: The Position Page screen of the EOS

The EOS graphical displays (or pages) were developed with the Builder Xcessory (BX) GUI editor developed by Integrated Computer Solutions (ICS) Inc. The main emphasis of this approach was to provide fast prototyping of new EOS pages. This was achieved by customizing the BX editor to become an EOS development editor (called EOS_bx). The EOS_bx is linked with the ACFS shared memory (or its own local version for isolated development). A set of customized Motif widgets were integrated with BX. These widgets have predefined properties and a small set of fields targeted for the ACFS-EOS environment. The ACFS EOS widgets are as follows:

- EOS Simple Button
- EOS Arrow button
- EOS Form
- EOS Set Value
- EOS Temporary Key Pad
- EOS Scale
- EOS Page

The EOS process is a conventional Xwindow process so it can be "pushed" and "popped" with other graphical windows as well as scaled to different sizes.

The EOS screen layout consists of four main areas; the Page Area (center and upper left), the Footer (along the bottom), the Status Window (upper right), and the Message Logger (below the Status Window). An example EOS screen is shown in Figure 5.

The Page Area is the main window of the EOS. It contains buttons and displays that allow the operator to call up different pages, input values, and monitor values. There are many pages in the EOS that can be selected and appear in the Page Area. The Footer, also referred to as the Hot Button Area, displays buttons which are always accessible, regardless of what page is displayed in the Page Area. Buttons here indicate if they cause an action, such as starting the FMS, or call up a page, such as Environment. The Status Window is a dynamic display of aircraft and environment conditions. What is displayed in the Status Window can easily be changed for each experiment. The Message Logger window provides important system or simulation messages to the operator. The EOS currently has pages supporting the following functions:

- Manual Reposition: Allows the operator to position the aircraft by entering latitude, longitude, heading, altitude, gross weight, and fuel weight.
- Initial Positions: Several "canned" positions, including aircraft configuration.
- Environment: Allows control of weather conditions, turbulence and visibility.
- Visual Control: Allows control of the OTW visual system.
- Preset Weather: Allows control of visual scene ranging from clear to zero/zero conditions.
- Aircraft Setup: Controls aircraft configuration and ground facilities.

Additional pages are created for each experiment's specific requirements.

Buttons on the EOS are color coded to indicate their type of action. Blue colored buttons call up new pages, pop-up windows, or input value windows. Pink colored buttons cause a specific action to take place. Pink button actions that take place include Screen Dump, Freeze, starting and stopping the FMS and ATC link, and Reset. Some of the pink action buttons also can be a different color to indicate a different current status. These specialized buttons are as follows:

- FMS: The color of the FMS action button indicates the state of the FMS. When the FMS is active, the button is yellow. When the FMS is not active, the button is pink.
- ATC: The color of the ATC action button indicates the state of the ATC link. When the ATC link is active the button is yellow. When the ATC link is not active, the button is pink.
- Reset: This button sets the simulator back to the last entered Initial Position (IP), regardless of whether the IP was entered manually or preset. The button turns yellow during the reset and back to pink when the reset has finished.
- Freeze: This button toggles the Simulation in and out of the Freeze State. The state of the Simulation is displayed with text and color on this button. Freeze on is indicated by red and the words "Freeze On". Freeze off is indicated by pink and the words "Freeze Off".
- Exit: Activating this button exits the EOS utility program.

Buttons are activated when contact is released, not initiated. If a button is accidentally pressed utilizing this technique, the operator's finger can slide contact to the side and disarm the button without activating it. Also, the operator can prepare for activation by pushing the button then watch some other screen for information on when to activate. The operator then does not need to look at the EOS screen, he simply releases his finger from the screen to activate the function.

COMMUNICATION TO SUBSYSTEMS

Communication with most simulator subsystems is done with UNIX socket procedures over Ethernet lines. This includes the communication with eight flight display computers, the sound generation computer, the EOS computers, other simulators (the ATC simulator, Boeing 747 simulator, or other simulators not co-located), and several additional devices. Standard TCP/IP communication protocols have been employed in order to simplify support for communications with the wide variety of platforms that are used on the ACFS, and to provide an easily supported connection capability to research systems.

Different TCP/IP packet protocols are used for different devices. The TCP packet protocol is used for asynchronous stream devices such as the touch screens and alpha-numeric displays. The User Datagram Protocol (UDP) packet protocol is used for the large synchronous data packets that drive the flight displays. Multicast UDP protocols are used for the Primary Flight Displays (PFD) due to the large amount of shared data used for the Captain's and the First Officer's displays. This allows both processors to receive the same data and reduce the amount of real-time network traffic. UDP packet protocol has the advantage over TCP/IP packet protocol of being much lower overhead on the processor due to the connectionless nature of the communication. Although UDP does not have as low an overhead as "raw" protocols, it does have the advantage of being widely accepted. This eases development by insuring documentation and debugging tools are available and simplifies routing to other networks to meet specific experiment communication requirements. It has been determined that the overhead difference between UDP and raw protocols is minor for the ACFS application.

By utilizing standard UNIX socket communication, virtually the same code can support both real-time experiment operations on the Challenge computer and simulation development on individual workstations. The remote processes (such as the flight displays) can run on the local development station using the same socket for internal communication rather than over an Ethernet line. Although the processes run slower in the development environment, the performance is usually adequate for developing or debugging new systems.

Communication with the OTW visual system and the simulator I/O device driver (a Singer-Link AST device for reading and writing to analog systems) is performed utilizing VME boards emulating a DR-11W interface. This is required as these systems are old enough that they do not support Ethernet communications.

SIMULATION ENVIRONMENT

SIMULATION OPERATION

Loading, starting and stopping the simulation is accomplished through graphical menus that execute script files. The same executable can be used in a variety of configurations and "customized" by the setting of environment variables and run-time parameters.

The normal procedure is to log into the ACFS account from one of the SGI Indy EOS workstations. A Motif compliant ACFS Simulation Control window appears and provides a list of the currently available simulator bases and simulator tool options for subsystems such as the FMS and for development tools. Desired options are selected and the Execute button activated to initiate the start-up operation. It requires approximately 30 seconds for the simulator to automatically load the software to all subsystems from the file-server or local disk and start up the entire simulation.

Once the simulation software is loaded the simulator is always in one of the following modes:

- Reset
- Freeze
- Freeze Off (run)

When the simulation has been reset, the simulation is returned to an Initial Position (IP). The desired IP can be selected from preprogrammed options available on the EOS Position Page. The IP may be on the ground at many different locations or in flight at some desired location, altitude, airspeed, etc. The Freeze mode stops a majority of the modules (aircraft, environment, data collection) and leaves the simulation in what appears to be a state of suspended animation. During Freeze, many system and I/O modules continue to function to allow the operator to inspect and control the simulation. All normal simulator operation continues when the Freeze is de-selected.

All simulator control functions and unique experiment functions are controlled from one or more of the EOS pages. The termination of the simulation is accomplished by selecting the Exit EOS button at the bottom right of the EOS display and then selecting Stop on the ACFS Simulation Control screen.

SIMULATION DEVELOPMENT

Simulation development is performed on one of three platforms. For initial development of an experiment, the engineer runs a version of the ACFS software (called the mini-ACFS) on a desktop SGI workstation. The mini-ACFS and full simulator have nearly identical software. The main aircraft model, FMC, and EOS are essentially identical. The communication to the flight displays is the same, but the socket connects to an abbreviated version of the display software that is running on the local workstation, rather than over an Ethernet line to the display computer for the cockpit. A few additional mini-ACFS specific control and display tools are available to make up for the lack of cockpit hardware availability. All these processes run in the one processor in the user's desktop machine instead of being distributed in the Challenge computer and over Ethernet lines in other computers. The system does not provide the process isolation and clock timing necessary for real-time and the limited processor capability limits the system to running slower than real-time. However, this is generally adequate for initial experiment development.

As the development process progresses, the engineer moves to a more powerful integration and test station that has several video display screens (vs. the one on his/her desk) in order to verify the functionality and interaction of the new system with all the ACFS software subsystems. Again, the development station utilizes essentially the same code that is used on the full mission simulator, only differing in some control and scheduling routines. Due to the increased screen space, this configuration can run the exact flight display software. As with the desktop mini-ACFS, the test and integration station does not run real-time, however this is generally adequate for most debug and test procedures.

For final test and verification, the entire ACFS is utilized. This provides the exact environment that will be used for experiment operations. This is particularly useful for tuning scenario code and event timing for experiment-specific operations. Following this phase, experiment operations and data collection takes place. Figure 6 shows a diagram representing the development, integration and test, and real-time facility interface.

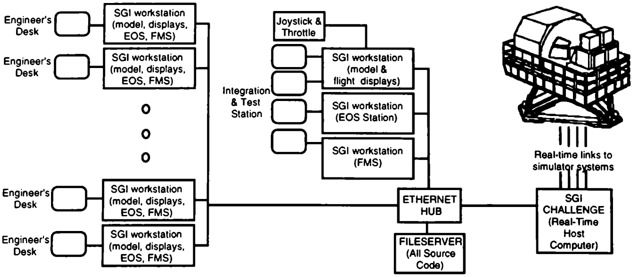

Engineer's Desk
SGI workstation (model, displays, EOS, FMS)
Engineer's Desk
SGI workstation (model, displays, EOS, FMS)
o
o
o
Engineer's Desk
SGI workstation (model, displays, EOS, FMS)
Engineer's Desk
SGI workstation (model, displays, EOS, FMS)
Joystick & Throttle
Integration & Test Station
SGI workstation (model & flight displays)
SGI workstation (EOS Station)
SGI workstation (FMS)
ETHERNET HUB
FILESERVER (All Source Code)
Real-time links to simulator systems
SGI CHALLENGE (Real-Time Host Computer)

Figure 6: ACFS development, integration & test, and real-time interaction

DEVELOPMENT TOOLS

For any stage of experiment development or operations most of the same tools are available. For a complete suite of debugging tools, the user compiles in debug hooks (a compilation flag) and then invokes the SGI CaseVision tool set. This tool set includes a Graphical User Interface (GUI) front end to the debugger, a static code analyzer, complete interactive help and documentation, and much more. The primary tool used for ACFS development has been the CaseVision Debugger (CVD). CVD is a modern graphical-based debugger providing a high-level language-based source code display and complete features such as break-points, single step, display and modify. Although CVD extacts a small performance penalty, the impact when used in real-time for process inspection has been negligible. Use of the tool in this way does not cause frame over-runs.

If the user wants to simply inspect and deposit into application code variables, he invokes an in-house built tool called the Global Common Utility (GCU). This tool provides access to all Global Common variables (which includes most parameters users wish to inspect or change).

In addition, there are other tools available for plotting and analysis. The current plotting process uses a predefined set of Global Common variables edited into an ASCII file. A plotting data file is written to during the real-time experiment run. The public domain plot utility PLOTMTV is used to plot and print the collected data.

CONFIGURATION MANAGEMENT

Software modifications and upgrades to the ACFS are made frequently to support the ongoing human factors research. Some changes to the simulation software are retained to become part of the baseline, others are temporary for a given experiment and are removed and archived. At any one time there may be up to a dozen engineers working on various experiments or doing software maintenance for the ACFS. However, to ensure integrity of the simulation during actual experiment operations, all development activity is isolated from the real-time systems. A rigorous configuration management system is required in order to track these continuous software and data modifications efficiently. To meet these constraints and provide a productive development environment, all source files and data are maintained on a file-server. A SUN Sparc 10 computer system with dual CPUs, 96 Megabytes of memory, and 7 Gigabytes of disk storage provides the Network File System (NFS) support for the engineering workstations. Development activity on the ACFS host computer system uses the NFS disk system. Real-time simulation loads can be run directly from the file-server. Tested "experiment-ready" real-time executables and data files are copied to ACFS host local disks to ensure isolation from the development environment during experiment operation.

The simulation software is maintained in multiple directory trees representing a logical decomposition of the flight simulator systems. Related software modules including their support files and data structures are defined as a Computer Software Configuration Item

(CSCI). The UNIX Source Code Control System (SCCS) is used to maintain revision level changes at the source code and data level. In the current system, a simulation development project is accomplished by doing the initial development in a minimally controlled base. As work progresses, the software is migrated to a moderately controlled base for system integration. Finally, the software is validated in an experiment base which is then frozen for the duration of the experiment.

PERFORMANCE AND PRODUCTIVITY IMPROVEMENT

The improvement in processor performance over the older system was dramatic, approximately one order of magnitude for each processor (with potentially 12 processors available). Memory availability is much more than required and hardware reliability has been 100%.

Improved productivity was gained in several ways. First, through the use of COTS hardware and software the facility does not need to maintain as large a staff of software personnel devoted to supporting an in-house real-time operating system. This frees up more of the software staff to develop research-specific software. Additionally, SGI provides a state of the art software development environment. The key elements of this environment can be utilized with the real-time OS, so the user is not required to develop and test his software in a non-real-time environment and then port it to the real-time system for re-testing. Also, the user does not need to learn and stay proficient at two or more different environments, and the system support staff does not need to support a unique (often in-house developed) real-time system. The improved toolset and the use of the same toolset for development and real-time has increased productivity.

Productivity is also significantly improved through the use of many mini-ACFS systems. With minor modifications to the simulation startup and schedule, the entire simulation can run on small single processor SGI machines in non-real-time. This provides a system that NASA researchers can use on their desks for initial concept evaluation, as well as for ACFS engineers to implement changes, and do initial testing. This reduces the need for expensive duplicate simulators for development, or using dedicated time early in the development phase on the main simulator complex. Since the cockpit flight displays are driven by SGI computers, the same processes can run on the development computer concurrently with the model, again not in real-time, but fast enough for development of most new research systems.

FUTURE PLANS

The system, as developed, is successfully performing research experiments more reliably and with less development effort than the previous system, however, there are many improvements planned. Some of the improvements are listed here.

The EOS functions and asynchronous I/O are computed on the same processor as the UNIX OS. As new processor boards are added, these functions will be moved to isolated processors to ensure appropriate response time.

The EOS will have several enhancements. The primary improvement will be a map display to provide the operator with better situational awareness. A real-time graphical EOS data plotter will also be added. Different COTS systems will be periodically evaluated and may be considered as replacements.

Current message logging has a simple priority and message structure. A flag based priority system would provide the operator a cue to important error, warning, and informational situations as they arise. This kind of message system is similar to the Engine Indication and Crew Alerting System (EICAS) messages used in the aircraft itself.

SGI provides a "frame scheduler" capability to provide synchronization control of processes distributed across multiple processors. The frame scheduler also provides access to a much higher resolution clock (21 nanosecond resolution). As new advanced subsystems are added to the ACFS simulation or higher frequency calculations are needed, the SGI real-time frame scheduler will be implemented.

An improved GCU is planned to allow more flexibility for inspection of variables in the main executive, MODSCH. The current GCU only allows predefined data files to be used. The new GCU will allow real-time editing of the variable list being used, as well as changing the display format, and browsing through each of the Global Common sections. A further enhancement should allow the changing of local variables as well as Global Common variables.

Planned improvements to the Software Configuration Management System include the ability to modify development configurations from the engineer's desk top with complete isolation from other simulation user activities by utilizing specialized UNIX scripts and MAKE files.

The integration and test computer is currently a single processor SGI workstation with two graphics displays. An SGI Onyx would provide the ability to closely model the multi-processor architecture that the host platform has while also supporting more displays. These displays could be used for a simulated OTW display, more flight instruments, as well as operator/developer interaction. This should greatly improve integration and test efficiency.

In addition to the computer system enhancements listed above, there are three major hardware improvements planned. A new high quality OTW Visual System and a new I/O system will be installed in the ACFS during the next year. In addition, a commercial Heads-Up Display (HUD) system, similar to current systems gaining popularity with several of the airlines, is being considered for installation in the ACFS.

SUMMARY

A multiprocessor implementation of a very complex full-mission research flight simulation using essentially Commercial Off The Shelf (COTS) hardware and system software is described. The real-time system has provided a significant performance and reliability improvement. The development and real-time implementation has provided common environments between the multiprocessor real-time computer, integration and test station, and desktop development systems. This common development, test, and operations environment saves resources in supporting many different platforms, physically moving files around, and user concern over data control. The use of common control and debug tools on all systems means there is no lost productivity by users being forced to learn new systems or move between different systems. There are several improvements planned to the simulator hardware and software to increase efficiency of development and operations even further.

References

Builder Xcessory 3.5 User's Guide
Integrated Computer Solutions Incorporated
Copyright 1995

Builder Xcessory 3.5 Reference Manual
Integrated Computer Solutions Incorporated
Copyright 1995

Advanced Concepts Flight Simulator
Programmer's Manual and ACFS Procedures
PM-020 Last Revision May 16, 1996

SGI IRIX On-Line Document Set
IRIX 5.3

AIAA-96-3514-CP

TACTICAL ENVIRONMENT DESIGN FOR AN ANTI SUBMARINE WARFARE HELICOPTER TRAINER

Antoinette Labbiento, Eng.
CAE Electronics Ltd.
St-Laurent, Quebec, Canada

Abstract

In tactical naval training applications, the simulated environment with which the training crew interact is a critical element in ensuring overall training effectiveness. The degree of realism, ease of use and instructor system controls permit the trainees to collaborate with allies and to counter hostile forces in order to ensure that the training objectives are met. Computer generated forces, allied and hostile, stimulate the ownship sensors and equipment to provide the trainees with the information required for training and tactical decision making in the anti submarine warfare role.

Introduction

CAE's Interactive Tactical Environment Management System, (ITEMS^{TM}), has been expanded to include an Anti Submarine Warfare Naval Environment. This includes an acoustic propagation environment; naval vehicles such as submarines, ships and helicopters with a dipping sonar capability; systems such as hull mounted sonars, sonobuoys and communication buoys and weapons such as torpedoes and depth charges. Additionally, tactics involving vehicle collaboration are simulated to counter the threat; the silent lurking enemy submarine.

ITEMS is a real time system for application in simulation, training and research and development. ITEMS simulates a real world tactical environment, which consists of atmospheric conditions, terrain and gaming area definition and computer generated forces. Computer generated forces are simulated with regards to their dynamic capabilities, on-board systems and behaviour. Vehicles may interact with one another or with human in the loop simulations.

This paper discusses the issues involved in the tactical environment design of an anti submarine warfare helicopter trainer through naval environment extensions to the existing ITEMS systems. A description of the ITEMS concept is presented along with a description of the naval application. The discussion of the naval application provides a basis for establishing the simulation elements to be developed and integrated.

Training requirements for naval helicopter crews demand an additional level of simulation beyond traditional flight training. These include: ship based operations, including take-offs and landings; operating as part of a helicopter formation at sea; providing command decisions and collaborating with other units to successfully accomplish the mission objectives. To this end, a detailed six degree of freedom ship motion model and a sophisticated companion aircraft system have been developed in addition to the ITEMS system to satisfy the needs of high fidelity training. Other typical training missions include: searching for submarines, weapons delivery and area surveillance.

The future potential for this technology is to expand it for use in other types of simulation environments including shipboard and submarine based trainers. This also includes the ability to provide the training environment in networked simulations using the Distributed Interactive Simulation (DIS) protocol, and to provide a tri-service training environment.

ITEMS Overview

The following provides a description of ITEMS current capability:

The Interactive Tactical Environment Management System (ITEMS) simulates a tactical environment based on the simulation of physical entities, i.e. players. Players are those elements that the ownship encounter during its training mission. Players currently include air, surface ships and ground platforms. Players are assigned to a team; friendly, hostile or neutral. ITEMS models players with respect to their dynamics capabilities, their equipment; sensors, weapons and communications and their behaviour. Behaviour enables player interaction with one another, with the environment and with the ownship trainees.

The simulation environment and training mission is defined in ITEMS by the tactical scenario. The tactical scenario is defined off-line and is controlled in real time by the user. Tactical scenarios contain all the information required to execute a tactical environment during the training exercise. This includes players, player behaviour actions and the external environment.

Scenarios are defined, off-line, before training commences, using a user friendly graphical scenario creation tool, Database Management System (DBMS). DBMS is a set of specialized data entry editors based on the X windowing system. These editors are used to enter all of the data to represent a complete tactical environment. Tactical scenarios are created off line and are down loaded to the simulation host for real time execution.

Scenario design is hierarchical. The user first creates the lower level libraries such as radar systems. Lower level entities are then combined to create a representative player model such as a fixed wing aircraft. In the scenario, players are assigned specific behaviours and are placed together in a common terrain gaming area with other players, DIS networked remote players, the ownship and the physical environment. All of these entities will interact with one another during the execution of the training mission. All of this information is stored in databases managed by DBMS. During scenario design, the user provides ITEMS with the information required to simulate the tactical scenario.

A crucial part in the creation of a realistic tactical environment is the modelling of player behaviour and reaction. This is based on the application of specific military tactics. ITEMS makes use of an expert system rules inference engine so that the information is determined by the user and not by the simulation software. The user determines player behaviours by the entering of rules. Rules are used by ITEMS players to determine their primary threat and also to execute their mission against this primary threat. Rules enable the definition of the rules of engagement in multi-sided conflicts. Rules are executed during the training session to provide realistic and interactive responses to the trainees or instructor role play actions. Rules enable ITEMS players to carry out tasks of an expert nature thereby providing tactical capability to players.

A major advantage to having tactical data defined by the user via DBMS is issues related to security. ITEMS is unclassified software. Databases that are created by the user may contain secure data but this data is not embedded in the software and can be stored separately taking into account all necessary classification regulations.

With the use of ITEMS, a tactical environment in which players interact realistically is provided. If desired, the tactical environment may be networked via the Distributed Interactive Simulation (DIS) protocol, to both manned simulators and to other tactical environments.

ITEMS has been used, up to this date, primarily in air force and army applications and has been developed with the requirements of these environments in mind. ITEMS is currently being upgraded to contain a full naval environment to be utilized initially in conjunction with rotary wing ASW helicopter training simulators. Next, the paper discusses the features which are being designed as part of the naval environment. These features are designed with respect to the needs of naval aviator, sensor and tactical operator training.

Naval Environment

The ITEMS Naval Scenario is depicted in Figure 1. This defines the major elements in the naval environment. As described in the introduction, this figure illustrates the hierarchical nature of the tactical scenario. The scenario consists of players who are assigned a mission route, initial position & inventories and a mission to accomplish during the training session. Players are themselves composed of the lower level systems such as weapons and sensors and navigational abilities.

The naval scenario design addresses trainee participation in such primary missions as Anti Submarine Warfare (ASW), Anti Surface Warfare (ASuW) and Anti Air Warfare (AAW). Additionally, trainees can engage in Search and Rescue missions over land and sea & procedural and tactical formation flying.

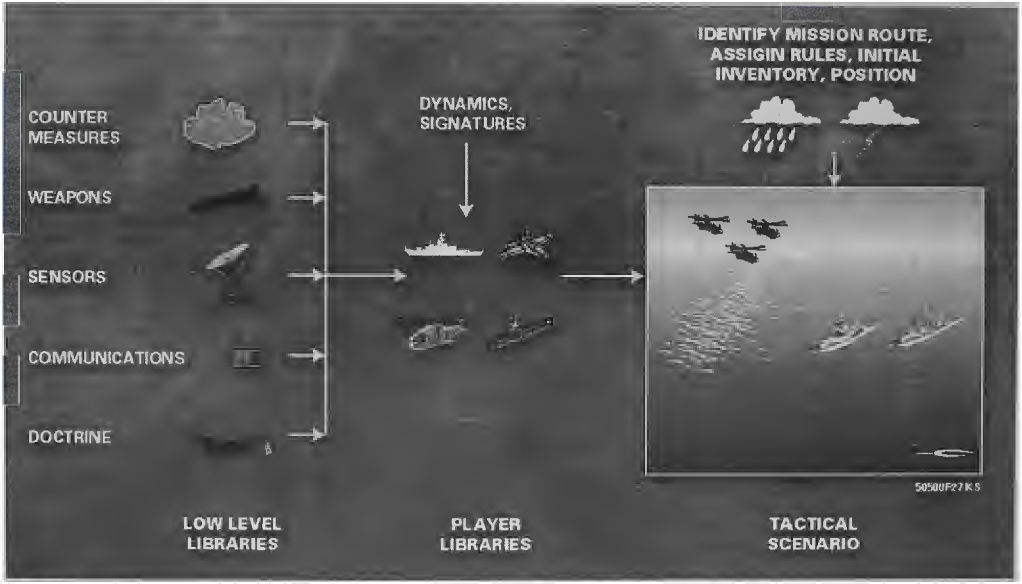


Figure 1 ITEMS Scenario Creation

In order to satisfy the above training objectives, new computer generated entities have been developed to simulate:

- Submarines
- Life Rafts
- Landing Ships
- Companion aircraft for formation flying.

These players are equipped with such newly modelled systems as:

- Active and Passive Sonar
- Sonobuoys
- Mobile and Static Acoustic Decoys
- Torpedoes
- Depth Charges
- Data Link
- Communication Buoys

Simulation of naval entities and sensor and weapon systems are primarily deterministic in nature, rather than statistical. However as these components will operate in a multi-player, multi-system scenario in real time during training, look up tables are utilized for acoustic propagation to effectively manage available computer resources. Also, ITEMS players are simulated using linearized dynamics, a more comprehensive six degree of freedom dynamics model is reserved for models which are required to perform specialized training tasks such as detailed ship modelling or support helicopters.

In order to combine these players and systems into tactical scenarios together with the own helicopter, behavioural and tactical responses are being provided such that the naval players will initiate conflict situations or react and behave appropriately in these situations.

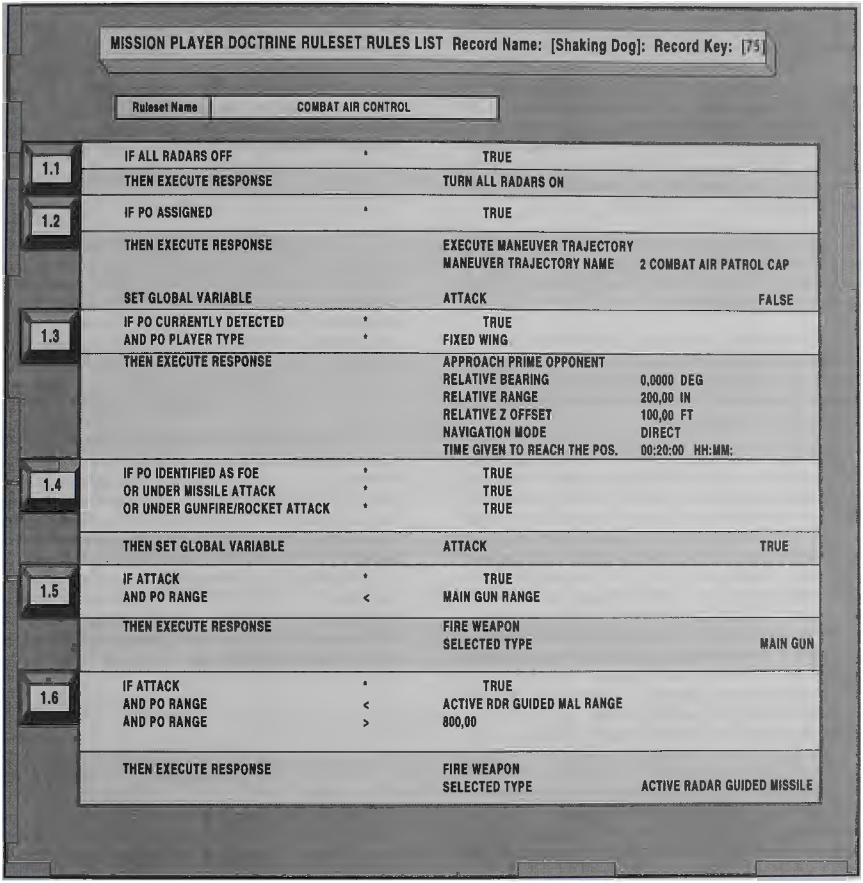
In ITEMS, as player actions and reactions are determined by user defined rules, the appropriate conditions and responses typical to the naval environment are being developed with subject matter experts. These include the relevant use of sensors such as sonar, radar and data link to build situational awareness, and the appropriate deployment of weapons and decoys in order to simulate both offensive and defensive actions as determined by current tactical doctrine. ITEMS provides the user the capability to initially develop these rules or doctrines and then to be able to modify them as needed during the simulator life as naval doctrine changes.

In Figure 2 there is an example shown of the types of rules that can be built to dictate a player's behaviour. The example is for a player to perform an air combat patrol.

In order to complement the naval environment features, automatic tactical manoeuvres for players are being developed to enhance the training realism and reduce instructor workload during the training session monitoring. These manoeuvres include zig zagging for ships, an auto dip pattern for naval helicopter with automatic dipping sonar , manoeuvres for units involved in a screening pattern and generic fixed wing and rotary wing search patterns for the purposes of submarine detection and prosecution.

During the training session, friendly ITEMS players will be operating in conjunction with the own helicopter or will result in situations were they will be under the direct command of the own helicopter crew. In order to close the loop in this situation, a comprehensive set of instructor controls are made available such that the instructor may effectively monitor ITEMS player actions and status. The instructor can also role play, override or take control of the ITEMS player to effectively carry out the intents of the situation commander, which in many cases will be the own helicopter crew observer, one of the trainees. These controls are available during the real time execution of the training session.

In addition, ITEMS provides a gods eye view display of the scenario during execution. This Tactical Situation Display (TSD) is a two dimensional map with terrain features underlay and with the player and own helicopter positions dynamically updated as required. Additionally,ITEMS provides a Forward View Display (FVD) which is a three dimensional view of the terrain and scenario. The viewpoint is operator selectable and can be slaved to a player position. The FVD is another useful tool for scenario monitoring and instructor interaction in the training session.



MISSION PLAYER DOCTRINE RULESET RULES LIST Record Name: [Shaking Dog]: Record Key: [75]

Ruleset Name	COMBAT AIR CONTROL

1.1	IF ALL RADARS OFF	*	TRUE	
	THEN EXECUTE RESPONSE		TURN ALL RADARS ON	
1.2	IF PO ASSIGNED	*	TRUE	
	THEN EXECUTE RESPONSE		EXECUTE MANEUVER TRAJECTORY	
			MANEUVER TRAJECTORY NAME	2 COMBAT AIR PATROL CAP
	SET GLOBAL VARIABLE		ATTACK	FALSE
1.3	IF PO CURRENTLY DETECTED	*	TRUE	
	AND PO PLAYER TYPE	*	FIXED WING	
	THEN EXECUTE RESPONSE		APPROACH PRIME OPPONENT	
			RELATIVE BEARING	0,0000 DEG
			RELATIVE RANGE	200,00 IN
			RELATIVE Z OFFSET	100,00 FT
			NAVIGATION MODE	DIRECT
			TIME GIVEN TO REACH THE POS.	00:20:00 HH:MM:
1.4	IF PO IDENTIFIED AS FOE	*	TRUE	
	OR UNDER MISSILE ATTACK	*	TRUE	
	OR UNDER GUNFIRE/ROCKET ATTACK	*	TRUE	
	THEN SET GLOBAL VARIABLE		ATTACK	TRUE
1.5	IF ATTACK	*	TRUE	
	AND PO RANGE	<	MAIN GUN RANGE	
	THEN EXECUTE RESPONSE		FIRE WEAPON	
			SELECTED TYPE	MAIN GUN
1.6	IF ATTACK	*	TRUE	
	AND PO RANGE	<	ACTIVE RDR GUIDED MAL RANGE	
	AND PO RANGE	>	800,00	
	THEN EXECUTE RESPONSE		FIRE WEAPON	
			SELECTED TYPE	ACTIVE RADAR GUIDED MISSILE

Figure 2 ITEMS Rules and Doctrine

Pilot Training

In reflection, ITEMS provides a comprehensive environment for tactical training situations. However ITEMS is a high fidelity simulation which provides additional features for specialized naval pilot training. These features include ship deck landing training under varying weather and sea state conditions in day or night. These features also include the procedures required to teach formation flying. Such training objectives were at one time not achievable in a training simulator environment. However with the continued evolvement of visual systems, these tasks have become suited and effective for simulator training.

In conjunction with MAXVUE, CAE's Image Generation System, ITEMS provides a highly realistic dynamic sea environment and highly representative landing ship and companion aircraft player models.

For specialized pilot training, a three dimensional ocean scene with the modelling of sea state, wind effects, ship wakes and rotor downwash are represented by the visual system. The modelling of sea states zero to six is represented with ship deck landing possible in all such states.

In order to accomplish this level of training,ITEMS provides an enhanced surface ship dynamics model, the landing ship. The landing ship is a six degree of freedom representative motion model which takes into account both sea state and the ship type. The ship is treated as a continuously distributed mass subject to a wave input at four equally spaced locations around the hull, two forward and two aft. The inputs give rise to buoyancy and damping forces which generate pitch, roll, and heave motion. Motion in the forward direction (surge) is through an autopilot system that maintains a constant ship speed. Sway motion is also represented. Motion in the yaw axis is represented by a first-order filter response. A control loop around this axis is used to maintain a demanded ship heading. This model provides for realistic motion cues to the own helicopter landing as it is flying the controlled approach to the ship.

Formation flying training requires the simulation of a comprehensive enhanced helicopter six degree of freedom dynamics model, the companion aircraft. The primary goal of formation flying training is to train the pilot to fly in formation with other aircraft and maintain proper rotor spacing. In order to accomplish this objective, the ITEMS aircraft is a virtual own helicopter "companion" with forces and moments generated at the main and tail rotor matched to the own helicopter performance.

These formations can operate with the own helicopter as either formation leader or wingman role. Typically, these helicopter formations consist of four members. The ITEMS companion supports different formation styles for in air and on ground, formation styles such as staggered, heavy and echelon. Also these formations take into account rotor spacings for day and night flight. The benefit that the virtual companion aircraft provides to formation training is related to one of training value for the given simulation cost.

DIS

In addition, the elements of the ITEMS naval environment are complemented via the use of the Distributed Interactive Simulation (DIS) protocol. ITEMS supports DIS and via this networking technology it is possible to link ITEMS based simulators to one another and to DIS networks to enable participation of ITEMS based simulations with other training devices.

Summary

In summary, the ITEMS naval environment is being developed with the needs of naval flight simulation training at the forefront. The potential of this technology spills over to ship or submarine operation room trainers and to naval tactical command training in general. All aspects of naval features are simulated. These include players, systems and tactics. In conjunction with the existing air, land and DIS networking functionality, the needs of intra-service applications may be studied as a future application for ITEMS. ITEMS is developed with the intent to fulfill the needs of high fidelity simulation training and it is clear that this goal will be promulgated throughout the many future naval environment applications.

References

1. Siksik, D.N., Beauchesne, G., Holmes R., "The Integration of Distributed Interactive Simulation Protocol Within an Interactive Tactical Environment", Interactive and Interoperable Simulation - The Challenges and Potential Benefits, The Royal Aerounatical Society, pp. 17.1-17.7, November 1994.

Acknowledgement

The author acknowledges the continued development of the ITEMS product by the dedicated and professional naval environment staff at CAE Electronics Ltd.

AIAA-96-3515-CP

OBJECT-ORIENTED FLIGHT-SIMULATOR TECHNOLOGY

S. Alagic[*], D. Ellis[†], R. Harder[‡], J. Hutchinson[§], G. Nagati[¶], and U. Rafi[#]
Wichita State University, Wichita, Kansas, 67260

Abstract

The fundamentals of a new, object-oriented software technology, for the design and implementation of flight simulators, are presented. The technology offers a powerful, general paradigm, and a methodology for complex structural and behavioral modeling. The underlying complex software required to support a sophisticated flight simulator is organized into an inheritance hierarchy of classes. Inheritance is used as a modeling technique, but at the same time, it allows software reusability. The main contribution of the paper is that it demonstrates that the object-oriented technology makes it possible to design and implement a generic flight simulator as a collection of general classes. This class library can then be used in such a way that specific simulators can be derived by inheritance from the generic one. The advantages of this approach are in major reductions in the required effort and cost. Yet another contribution of the paper is to demonstrate that the object-oriented database technology is a superior generic database support for the design, implementation and use of flight simulators.

I. Introduction

This paper proposes a technology that overcomes the following inadequacies of most existing flight simulator technologies:

- The absence of a general modeling paradigm that would meet the challenges of flight simulator technologies and at the same time have full support by a suitable computer technology.

- Usage of obsolete software technology, which does not promote modular design and reusability of the existing results.

- The absence of a generic database technology appropriate for flight simulators.

The underlying software system of a flight simulator is very complex. An object-oriented programming environment allows this support to be carefully organized into meaningful units encapsulating structural and behavioral properties. These units are in fact user defined types of objects and they are organized into an extensible hierarchy of classes.

The object-oriented paradigm supports information hiding[2] as a powerful technique for managing the complexity of the flight simulator software system. Information hiding is a technique in which the implementation details of a complex object are separated from its interface. The interface specifies the types of actions applicable to an object. The implementation details are hidden from the user and contain the "methods" or program code required to perform the actions.

The object-oriented paradigm also allows refinement and tailoring of existing solutions in a systematic and natural manner. Using inheritance, a generic and quite possibly crude flight simulator model represented by a collection of classes may be improved through a number of levels. The lower levels represent refinements inheriting from the upper, already existing levels. The inheritance hierarchy grows by adding more specific classes to the already existing hierarchy, reusing all the features of the existing classes[2].

Success of a particular flight simulator technology depends critically on its associated visualization technology. Object-oriented technology has already been successfully applied to advanced graphics and geometric packages on many existing platforms. The object-oriented paradigm allows these

[*] Professor, Dept. of Computer Science, Faculty Fellow of NIAR (National Institute for Aviation Research)

[†] Director of Research, NIAR, Associate Fellow AIAA

[‡] Instructor, Aviation/Computer Science, Hesston College

[§] Professor, Dept. of Mathematics, Director of Operations, NIAR

[¶] Associate Professor, Dept. of Aerospace Engineering, Associate Fellow AIAA

[#] Graduate student, Dept. of Computer Science packaged object libraries to be (re)used in a flight simulator application in the same way that a flight simulator object library could be later (re)used by other flight simulator developers.

A sophisticated flight simulator necessarily relies on a vast amount of data. Database management issues thus have a key importance. Unlike the traditional relational database model, the object-oriented paradigm allows modeling of objects which are complex both structurally and behaviorally[2,4]. An aircraft object, for example, has complex structure requiring appropriate representation of components, systems and geometric properties. At the same time, an aircraft object behaves according to complex behavioral patterns, given by a quite sophisticated aerodynamic model. The aircraft's responses to actuators (commands) or external forces (wind) illustrate this behavioral aspect of the object-oriented model.

In a truly general flight simulator we envisage extensive usage of object-oriented database management. For example, in order to provide training for the two most critical operations of takeoff and landing, at different airports and with different aircraft, an object-oriented database of airports and aircraft is required.

The object-oriented paradigm presents a different although more natural approach to traditional software design and implementation methods. The concepts of the paradigm will be illustrated in the following sections by presenting various aspects of the design and implementation of a flight simulator. It should be noted that as is often the case in design problems, many correct solutions are possible. As the model design progresses, the reader should note that the object-oriented paradigm in fact encourages the finding of these solutions through its real-world approach to modeling.

II. Complex Objects

At the conceptual level, complex objects are modeled in terms of has-a relationships. For example, an airport can be modeled in terms of has-a relationships as illustrated in Fig. 1.

Each one of the component objects of an airport is in fact a complex object itself, and can be decomposed further, as in Fig. 2.

This decomposition process proceeds until we reach objects of simple types. By contrast, aircraft_queue is a complex object obtained by instantiation of a generic, reusable object type Queue.

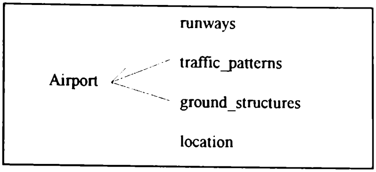


Fig. 1. Has-a Relationships

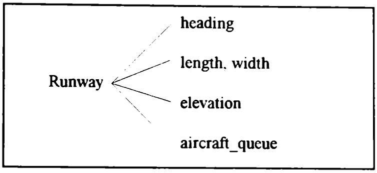


Fig. 2. Has-a Relationships

Airplane is a complex object which can be modeled in terms of has-a relationships as in Fig. 3.

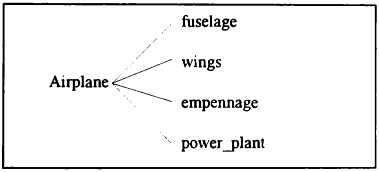


Fig. 3. Has-a Relationships

Components of an airplane are also complex objects, and may be decomposed further in terms of has-a relationships as illustrated in figures 4,5 and 6.

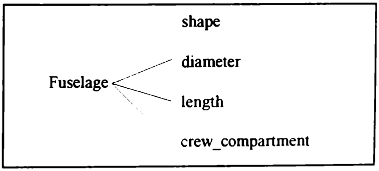


Fig. 4. Has-a Relationships

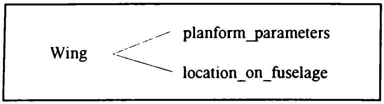


Fig. 5. Has-a Relationships

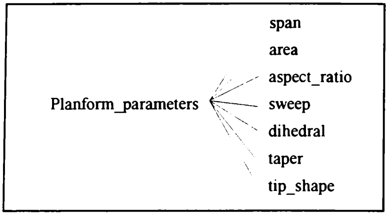


Fig. 6. Has-a Relationships

A key difference between the object-oriented approach and the conventional approaches is that object-oriented programming languages allow structuring of software components of a flight simulator according to the above conceptual and familiar views. We will represent has-a relationships along the horizontal axis of a conceptual model. A different type of relationships explained in the next section will be represented along the vertical axis of the conceptual model.

III. Inheritance

Has-a relationships can be represented in more conventional software technologies using complex (embedded) record structures. Representing is-a relationships using conventional technologies is much more difficult. An illustration of modeling the aircraft objects in terms of is-a relationships is given in Fig. 7. It is, in fact, an inheritance hierarchy constructed in accordance with the FAA classification.

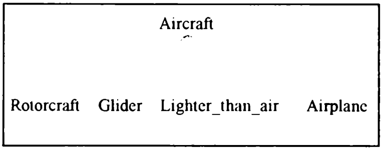


Fig. 7. Is-a Relationships

At the first level in the above diagram, we have a generic Aircraft object type, with four immediate subtypes, Rotorcraft, Glider, Lighter_than_air and Airplane. These four subtypes inherit all the properties of aircraft.

An illustration of some subtypes produced by further specialization is given in Fig. 8.

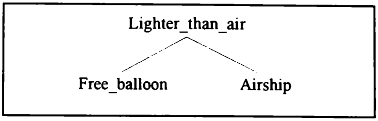


Fig. 8. Is-a Relationships

A further illustration is the inheritance hierarchy shown in Fig. 9.

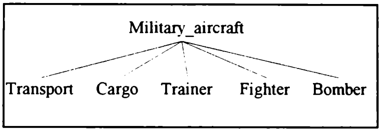


Fig. 9. Is-a Relationships

IV. Multiple Inheritance

Inheritance allows structuring the model into a specialization hierarchy. When a new class is added to the existing model, it specializes an existing class by adding specific properties (attributes) and specific behaviors, in addition to the inherited ones. It is often the case that a new class needs to be introduced in such a way that it combines the properties and behavior of the already defined classes. This situation is supported explicitly in the object-oriented model by a modeling technique called multiple inheritance.

Self-explanatory examples of multiple inheritance are shown in figures 10 and 11.

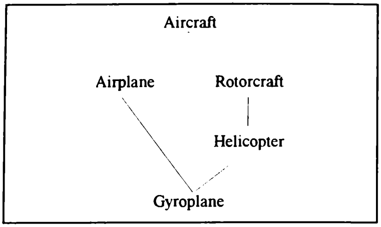


Fig. 10. Multiple Inheritance

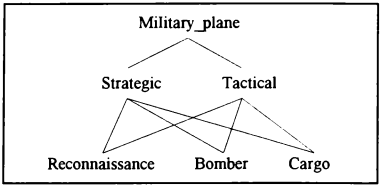


Fig. 11. Multiple Inheritance

V. Classes and Methods

Each of the object types in the above diagrams encapsulates not only structure, but also behavior. The structure is expressed in terms of has-a relationships and behavior in terms of methods. Such an encapsulated definition is called a class. An object is an instance of a class. It responds to messages that it receives, as long as its class has a method which specifies what the object's response to a message should be. Methods are thus recipes represented as functions.

A class interface consists of private and public components. Public components are typically specifications of types of messages that can be sent to objects of a class. Private components determine the state of objects of a class. The only way to affect the private components is by invoking public methods.

Inheritance applies to both structure and behavior. For example, a specific subtype of the aircraft type inherits all the structural properties of the generic aircraft type, but also all the generic behavior. In addition, a specific aircraft type has its specific attributes and its specific behavioral properties. Inheritance of behavior is complemented by the need to redefine it for a specific subtype.

Classes allow representation of a variety of types in the flight simulator environment, such as geometric types, images, and a variety of other types which are nonstandard for programming languages or database systems.

Changing the representation details would not affect the rest of the underlying software system, as it relies on the interfaces of the classes whose implementation details are being changed. Even major changes in the flight simulator model may be thus localized in the underlying software system, leading to a high-level of productivity in maintaining the already developed, complex software.

A paramount issue is further development and refinement of the already developed flight simulators. Such refinement is required for a variety of reasons: the results of testing, more sophisticated aerodynamic and air environment models, and more sophisticated aircraft.

A crude aircraft model is usually based on simple kinematic equations[3]. Such a generic model would in the object-oriented approach be represented by a collection of generic classes. Adding further complexity to the model would then require extending this hierarchy, with more elaborate classes refining the crude model with aerodynamic features.

VI. Control System Modeling

Object-oriented modeling of airplane aerodynamics requires specification of a class (Fig.12) which represents the control system. The interface of the Control_system class contains specification of the types of command input messages which correspond to moving stick, changing throttle, pressing pedals, changing flaps, etc. These command inputs affect the positions of ailerons, elevators, rudder and flaps. The actual positions are in fact private components of the state of the airplane. These components are affected by sending the messages move_stick, press_pedals change_throttle or change_flaps. The actual positions can be changed directly only in the implementation part of the control system.

In the private part of this class we have just one complex component which represents the airplane control surfaces.

The implementation part of the Control_system class (not presented in the paper) contains all the representation details of a control system which determine how the control inputs actually affect the positions of ailerons, elevators, rudder and flaps. This includes the effects of gearing, augmentation, etc.

```
Class Control_system {
public:
        void move_stick(float) ;
        void press_pedals(float);
        void  change_throttle(float);
        void  change_flaps(float);
private:
        Controls controls;
};
```

Fig. 12. Control_system Class

The information hiding principle allows hiding the representation details of a control system from the rest of the flight simulator system. This principle allows the representation details to be changed without affecting the rest of the flight simulator software. The design and implementation of the rest of the software system is based entirely on the Control_system class interface. It need not, and should not be concerned with the details of implementation of this class.

The public part of the interface of the class Controls (Fig.13) contains specification of the types of read-out messages which return the actual positions of ailerons, elevators, flaps, speed brakes, throttle, landing gear, etc. The other types of messages specified in the public part of the interface are those that affect the positions of these control surfaces. The private components consist of the actual positions of the control surfaces.

Yet another layer of sophistication requires modeling of systems such as the engine control system or the landing gear system, to mention a few component systems of the airplane model. Each one of these is in the current practice approached first through a simplified (called equivalent) model[3], which can be subsequently refined using the methodology described above.

Further development of an aircraft (not necessarily airplane) model would involve classes representing intercoupling of the airplane model with the air environment and the ground environment models[3]. Of course, the latter two would also be represented as hierarchies of classes. In prototyping or composing a flight simulator model, each one of these hierarchies can be accessed at the chosen level of modeling detail. This gives the level of flexibility that can hardly be accomplished in other approaches.

```
Class Controls {
public:
  float get_elevator_deflection();
  float get_aileron_deflection();
  float get_rudder_deflection();
  float get_flap_deflection();
  int get_speed_brake_deflection();
  int get_landing_gear_position();
  float get_throttle_position();

  void change_elevator_deflection(float);
  void change_aileron_deflection(float);
  void change_rudder_deflection(float);
  void change_flap_deflection(float);
  void change_speed_brake_deflection();
  void change_landing_gear_position();
  void change_throttle_position(float);

private:
   float aileron_deflection;
   float elevator_deflection;
   float rudder_deflection;
   float flap_deflection;
   int speed_brake_deflection
   int  landing_gear_position;
   float throttle_position;
};
```

Fig. 13. Controls Class

VII. Aerodynamic Modeling

In order to illustrate how object-oriented technology applies to aerodynamic modeling, we will base our exposition on the approach to aerodynamic modeling of fixed_wing aircraft presented in[5]. Aerodynamic forces and moments acting on an aircraft are specified in the class Dynamics (Fig.14).

The implementation part of this class contains details of computation of the aerodynamic forces and moments. All of them except for weight_force and thrust_force depend upon the wing area and dynamic pressure, where the latter is computed from the air density and aircraft velocity.

Computation of the lift force is in addition based on the angle of attack, elevator deflection and flap deflection. Computation of the drag force is based on the angle of attack, flap deflection, landing gear position, and speed brake deflection. Sideforce is computed from the sideslip angle, roll rate and yaw rate.

```
Class Dynamics {
public:
  float lift_force();
  float drag_force();
  float side_force();
  float roll_moment();
  float pitch_moment();
  float yaw_moment();
  float weight_force();
  float thrust_force();
private:
  Observers observers;
  Controls controls;
  Geometric_properties geometry;
};
```

Fig. 14. Dynamics Class

Computation of moments follows similar rules. The roll moment depends upon the wing span, aileron deflection, rudder deflection, sideslip angle, roll rate, yaw rate, and true airspeed. The pitch moment is computed from the angle of attack, elevator deflection, flap deflection, pitch rate, angle of attack rate, mean aerodynamic chord and true airspeed. Computation of the yaw moment is based on rudder deflection, aileron deflection, sideslip angle, yaw rate, roll rate, wing span and true airspeed.

All the details of the actual computation rules are completely hidden from the rest of the flight simulator software system. This makes it possible to change the rules without affecting the rest of the software system. There are several important cases in which such a change may happen. One of them is choosing a different aerodynamic model. The other one is adaptation of the computation rules to a particular type of aircraft. A third case is extending a generic aerodynamic model for steady state flight conditions and small perturbations with nonlinear behavior of the forces and moments which occurs with major changes of the relevant variables. Adding sophistication to an existing model leads to a simulator which offers a full spectrum of flight conditions, from altitudes that range from sea level to service ceiling for each particular aircraft, and for every conceivable combination of configurations and control settings.

The private part of the class Dynamics consists of three complex components. One of them represents the aircraft control surfaces as explained in the previous section. The class Observers (Fig. 15) captures a collection of observable properties relevant to the computation of the aerodynamic forces and moments.

```
Class Observers {
public:
  float angle_of_attack();
  float angle_of_attack _rate();
  float sideslip_angle();
  float roll_rate();
  float  pitch_rate();
  float yaw_rate();
  float air_speed();
  float air_density();
};
```

Fig. 15. Observers Class

The third private component encapsulates geometric properties of an aircraft relevant to the computation of the aerodynamic forces and moments.

```
Class Geometric_properties {
public:
    float wing_span();
    float wing_area();
    float mean_aerodynamic_chord();
private:
    float wing_span;
    float wing_area;
    float mean_aerodynamic_chord;
};
```

Fig. 16. Geometric_properties Class

The final class interface that we present in this paper corresponds to the equations of motion (Fig. 17).

```
Class Motion{
public:
  float acceleration_X();
  float acceleration_Y();
  float acceleration_Z();
  float roll_acceleration();
  float pitch_acceleration();
  float yaw_acceleration();
private:
  float inertia;
  Dynamics dynamics;
  Geometric_properties geometry;
  Observers observers;
};
```

Fig. 17. Motion Class

The public interface contains specification of messages that return three linear and three angular accelerations. The private part consists of components that are needed to compute these accelerations.

VIII. Polymorphism

Software reusability in the object-oriented technology is based on inheritance and genericity. Inheritance has been discussed. Genericity means that classes can be defined as templates equipped with a type parameter. Such a class defines a family of classes, one such class per a specific class substituted for the type parameter. This feature is known in programming languages as parametric polymorphism. For example, typical parametric (generic) classes are given in Fig.18, where T stands for any class.

```
Set<T>
List<T>
Array<T>
Queue<T>
```

Fig. 18. Generic Classes

The generic classes in Fig. 18 can be instantiated as in Fig.19.

```
Set<Ground_structure>
List<Runway>
Array<Aileron>
Queue<Aircraft>
```

Fig. 19. Specific Instances of Generic Classes

An example of usage of a generic (parametric) class is given in Fig.20.

```
Class Runway {
public:
   int enqueue(Aircraft);
   int dequeue(Aircraft);
   void display_queue();
private:
   int heading;
   float length, width;
   float elevation;
   Queue<Aircraft> queue;
};
```

Fig. 20. Using a Generic Classe

A private component of an object of Runway class is a queue of aircraft. It illustrates another kind of polymorphism in object-oriented systems. A runway queue may naturally contain aircraft of any specific type. Thus objects of any class derived by inheritance from the class Aircraft may be placed in an aircraft queue. Although this is what one would naturally expect, supporting it safely in a programming language environment is typical only for object-oriented languages.

IX. Airplane Model

With all the apparatus introduced so far we are now in a position to specify the airplane model (Fig.21).

```
Class Airplane {
public:
    Control_system controls;
    Set<Instrument> instruments;
    Air_frame frame;
    Dynamics dynamics;
    Position position;
    Orientation orientation;
};
```

Fig. 21. Airplane Class

Classes Control_system and Dynamics have already been specified. A class Air_frame may be specified as in Fig.22.

```
Class Air_frame {
public:
   Wing left_wing, right_wing;
   Fuselage fuselage;
   Empennage empennage;
   Array<Engine> engines;
};
```

Fig. 22. Air_frame Class

Further decomposition of the above specifications follows the same methodology.

The classes Airplane and Air_frame present a different option in information hiding. This option makes the components of an airplane object and an airframe object available to users in read-only mode. This read-only exposure of object state allows queries which refer to those components. Object-oriented queries are explained in the next section.

X. Database Support

The requirement for appropriate generic database support is based on the following considerations.

In the design process, flight simulator design data must be appropriately managed. This includes storing the data, modifying it (updating), selecting and displaying.

A sophisticated, general type of flight simulator would allow selection of an aircraft type, a ground environment model (for example, a particular airport) and an air environment model.

The data required in the above activities must be persistent. Data is persistent if its lifetime extends beyond the lifetimes of processes in which the data is being created and used.

User-friendly query facilities, typical for database technologies, become a major novelty in flight simulator environments. These query facilities allow easy exploration of flight simulator data in the design process, selection of an aircraft type based on its properties, selection of a ground environment model (such as an airport), etc.

In all of this we have to bear in mind that the actual data belongs to a variety of types. Some of that data is numeric, but much of it belongs to complex types such as aircraft, wing, engine, airport, ground_structure, etc.

An example of a query is:

List all distinct wing spans of all airplanes with mean aerodynamic chord less than 6 ft and with the total power less than 750 kW.

In the syntax of OSQL[4], this query would be expressed as in Fig.23.

```
select distinct(a.frame.left_wing.span +
             a.frame.right_wing.span)
from Airplane a
where a.frame.mean_aerodynamic_chord  < 6
and  sum(select p.power
         from  a.engines p) < 750
```

Fig. 23. Object-oriented Query

Of course, a pilot is not expected to ask queries in OSQL. Rather, he/she would have a touch screen or a similar device for selecting a query whose implementation in OSQL is illustrated by the above example. Other examples of object-oriented queries are:

List all the airports within a 500 mile radius from the current aircraft position with runways longer than 8,000 ft.

List all the airports where an aircraft can land from its current position.

List all the runways along the planned flight path where an emergency landing can be made.

XI. Computing Infrastructure

The computing infrastructure for this project is based on the facilities of our Object-Oriented Research Lab. It consists of a four-processor SUN workstation, an SGI workstation, and 5 NCD X-terminals.

The object-oriented software includes compilers for object-oriented programming languages (two C++ compilers and Eiffel compiler), and two object-oriented database management systems: O2 and ODE. O2 is a commercial system, and ODE is a research prototype from AT&T Bell Labs.

The actual implementation has been carried out in O2. The graphic package that has been used is Open-GL.

XII. Conclusions

Our goal in this paper was to demonstrate that there is a lot to be gained if object-oriented technology is used for the design and implementation of flight simulators.

The first advantage that the object-oriented technology offers is a general modeling paradigm which is particularly well suited for both structural and behavioral complexity of flight simulators.

The second advantage is that this general paradigm has a sophisticated supporting software technology. The object-oriented software technology is superior to typical software technologies used in flight simulators. It promotes modular design and careful logical structuring of the complex underlying software required in flight simulators. These properties affect the most critical issue of software technology in general: software maintenance.

Perhaps the most significant advantage of object-oriented software technology is software reusability. A contribution of this paper is in demonstrating that it is in fact possible to design and implement a family of generic flight simulators organized into an inheritance hierarchy, with a generic flight simulator model at the top. Such generic models can then be adapted by specialization to produce specific flight-simulator models for particular aircraft.

This hierarchical development based on inheritance typically consists of a number of levels. The savings in the required time, effort and cost is significant in comparison with the currently used approaches. Although there is some amount of sharing of software for flight simulators at the present time, the difference is that the object-oriented technology has been developed with the idea of reusability as its cornerstone. Object-oriented technology offers a systematic technique for reusability, and lifts it to a higher level.

Yet another advantage of object-oriented software technology is a recent proliferation of advanced graphic packages which are naturally object-oriented and have the same general advantages of the object-oriented technology in the specific and extremely important area of visualization.

The final advantage of the object-oriented technology is that it offers a generic database technology which is particularly suitable for flight simulators. This advantage remedies two problems. One of them is that a generic database technology is typically not used for flight simulators. The second problem is that the database technology which dominates the market is in fact unsuitable for flight simulators.

Finally, we mention briefly that advanced techniques of object-oriented modeling relevant to the problems discussed in this paper are reported in our earlier paper[1]. The paper in fact presents the fundamentals of a typed and temporal object-oriented database technology. Our previous experiences with flight simulators at the time when we did not use object-oriented technology are reported in[6].

XIII. Acknowledgments

Suad Alagic acknowledges support from the National Institute for Aviation Research and from DOD (DEPSCoR) under the Army Research Office Grant Agreement No. DAAH04-96-1-0192.

The equipment of the Object-Oriented Research Lab used in this project has been funded by a DOD Research Instrumentation Award under Defense University Initiative, Grant No. DAAH04-95-1-0007.

Parts of the project have been implemented within S. Alagic's graduate course "Object-Oriented Databases", and a number of students offered interesting insights into the problem.

Bibliography

1. Alagic S., *A Statically Typed, Temporal Object-Oriented Technology*, Transactions on Information and Systems, IEICE, Vol. E78-D, No. 11, 1995.

2. Atkinson M., Bancilhon F., DeWitt D., Dittrich K. and Zdonik, S., *The Object Oriented Database System Manifesto,* Proceedings of the First Object-Oriented and Deductive Database Conference, Kyoto, 1989.

3. Barnes A.G., *Modeling Requirements in Flight Simulation*, Aeronautical Journal, Dec. 1994.

4. Cattell R.G.G. (ed*), The Object-Oriented Database Standard*: ODMG-93, Morgan Kaufmann Publishers, Inc., 1996.

5. Galloway, R.T., *Aerodynamic Math Modeling, Twelfth Annual Flight and Ground Vehicle Simulation Update*, Binghamton University, 1996.

6. McCauley, S.G. and Nagati, M.G., *PC-Based Expandable/Affordable Flight Simulator for Education and Entertainment*, AIAA Flight Simulation Technologies Conference, Baltimore, MD, August 1-3, 1995.

AIAA-96-3516-CP

A STUDY OF SEARCHING ALGORITHMS FOR REAL-TIME SIMULATION OF A DYNAMIC SYSTEM

Yoon S. *

Korea Institute of Aeronautical Technology
Korean Air
Seoul, Korea

ABSTRACT

In real-time simulation of a dynamic system such as an aircraft encountered are parameters whose discrete values stored in various forms of tables. A dynamic system is typically defined with ordinary differential equations. Thus parameters are required to be computed repeatedly in every integration frame. The process is called function generation which consists of searching and interpolation. There applies a special constraint to searching in case of real-time simulation, which comes from the fact that fixed integration steps are generally prefered in real-time simulation. Therefore, the worst case of iteration must be predictable in order for searching to be performed within a fixed time-frame. This study is to compare various searching algorithms available for real-time simulation of a dynamic system and to develop an effective one. The comparison of worst cases says that the simple bisection search is the most suitable for the purpose among algorithms used in the literature. A new concept called dynamic-window search is also developed in the paper, considering the relation between system dynamics and parameters.

INTRODUCTION

In real-time simulation we face many times such situations as to compute the function value of a concerned point in a given data table with discrete keys. The required process is called function generation and divided into searching and interpolation. Typically, function values are stored discretely for certain keys in a format of n-dimensional data array. In flight simulation most values of aerodynamic coefficients and engine parameters for the flight model are stored in 1- or 2-dimensional data tables. Each frame of integration for differential equations of motion includes scores of function generations. Even in real-time simulation of a simple aircraft it is very crucial to accomodate an efficient algorithm for function generation.

* Chief Research Engineer
Simulation Technology Center, AIAA Member

In real-time simulation of a dynamic system it is typical to encompass sorted parameter tables with even or uneven keys. The tables are ususally sorted in their natures. If not, they can be sorted off-line before real-time simulation starts. That is, only searching algorithms with respect to sorted data tables are concerned in this study. Traditional methods of searching n-dimensional sorted array include bisection, modified bisection[1], and interpolation[2,3] searches. There are some other efficient algorithms claimed to be efficient such as fast search[4] and padded lists[5].

Since searching algorithms require unkown number of probes in general, real-time simulation concerns worst cases of iterations which must be counted in the frame size of integration used for simulation of dynamic systems. In this paper description of various searching algorithms widely used with sorted data tables is presented along with their comparisons. The comparison result indicates that simple bisection search is rather superior to more sophisticated searching methods available in the field.

If the data table is related to state variables of the dynamic system, it is possible not to search the entire table for a concerned location or a search key. A window, a portion of the table, can be generated where the search key resides by considering the dynamics. The window becomes smaller if the prediction of data dynamics is more accurate. The window moves from time to time throughout the simulation, and its size may also vary. It is another way for fast search, which is very practical for real-time use.

SEARCHING

There are a bunch of searching algorithms available in the field. In this paper only a few of them, such as bisection[1], interpolation[2,3], and fast[4] searches, are considered since data tables can be sorted off-line beforehand. It should be also understood that search is not necessary in the case of a sorted data table with even keys, since the data value divided by the fixed interval size between two consecutive points reveals the key. Therefore, search holds meaning with unevenly distributed keys.

Bisection search is a method to find a zone with a concerned data point by starting at (m/2+1)-th position and halving intervals in a given data table with m+2 keys from 0 to m+1. Thus this method locate the zone containing concerned data after $\log_2 m$-th probe in the worst case[2].

Suppose it is known that y, which is to be found, is located on an interval (x(i), x(j)). In interpolation search the next value accessed will be x(k), where k is found from the formula

$$k = (j - i)(y - x(i))/(x(j) - x(i)) + i + 0.5$$

If k = i, k is increased by 1, and decreased by 1 if k = j. If y = x(k), then the search is finished. Until y is located between x(k) and x(k+1), this searching process is repeated by resetting the interval for (x(i), x(k)) or (x(k), x(j)). Its worst case is to locate the concerned data point in O(m) time[2].

Interpolation search is faster than binary search for a modest size ordered list of values from a uniform distribution[2]. There exist some other variations to the interpolation search. One variation called fast search[4] is to achieve the probe in $O(\log_2 \log_2 m)$ expected time, and claimed to be more robust. It requires probes of $O((\log_2 m)^2)$ time in the worst case. Another variation employs a special data structure, the pedded list[5], which amounts to about the same speed. However, it should be noted that comparison of the worst cases are more important than comparing average numbers of probes in real-time simulation. If iteration or probe number is not fixed in searching algorithms, each frame of updates must be prepared for the worst case in real-time simulation since the frame size is generally recommended to be fixed. $O(\log_2 m)$ of bisection search is even superior to $O(m)$ and $O((\log_2 m)^2)$ of interpolation and fast searches in the worst cases, which means we have no benefit from adding more complicated logics in real-time simulation.

Aforementioned searching algorithms still have a room for improvement in real-time dynamics simulation by adopting window concept. The size of a given data table can be made smaller if the data point we are looking for has certain dynamics. It is not necessary to search a whole database for the data point, because it moves around the position of the previous frame. It is possible for the initial position of the data point to be calculated before real-time simulation begins. The data point moves based on its own dynamics and the integration step size h used in the simulation. For example, assume that a table of $C_L(\alpha)$ is given and α is a state variable integrated at each time frame. The function $C_L(\alpha_i(t))$ at the i-th integration step is to be found, and Euler integration is applied to α. Then

$$\alpha_i = \alpha_{i-1} + h\,(d\alpha/dt)_{i-1}$$

If w is the varying interval size between two break points or keys, then the K_i-th interval in the table, where α_i is expected to reside at the i-th integration step, can be determined as follows:

$$K_i = K_{i-1} + (\alpha_i - \alpha_{i-1}) / W_{i-1} = K_{i-1} + (h/W_{i-1})\,(d\alpha/dt)_{i-1}$$

Here K_{i-1}, h, W_{i-1}, $(d\alpha/dt)_{i-1}$ and α_{i-1} are known. It is reasonable to use W_{i-1} for reference if intervals do not vary rapidly from interval to interval in a given table. Thus the neighbor of K_i can be predicted. A window where search is to be made can be determined depending on integration algorithm, step size h, and interval w used in the simulation. Off-line trial-and-error efforts will make the window size more suitable, which may vary frame by frame. The dynamic-window concept is illustrated in Fig. 1.

NUMERICAL TESTS

Numerical tests were conducted by applying the aforementioned searching schemes, which are bisection, interpolation, and fast searches, to various dimensional tables of various sizes. Test results indicate that worst cases were not encountered in most test runs. Interpolation and fast searches worked very efficiently in average as predicted from the references, while bisection

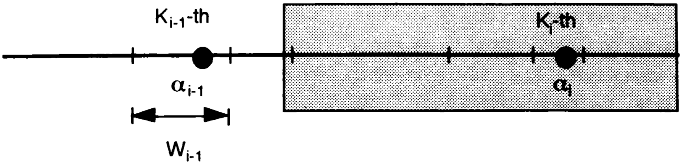


Figure 1. Schemetic diagram of the dynamic-window concept

search worked poorly in most cases. However, constraints of real-time requirements concern worst cases of the algorithms. The bisection search was always stable and reliable even if it was not efficient at all in average. The cost of computation burden in sophisticated search algorithms makes them even less suitable for real-time simulation.

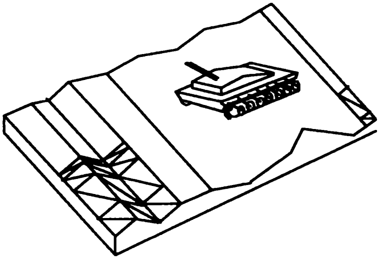


Figure 2. A military tank moving on 3-dimensional terrain composed of traiangular polygons

The proposed dynamic-window search was implemented in real-time simulation of military tank dynamics[6]. The dynamic model used in the simultion does not count the degrees of freedom of the road wheels, whose behaviors are determined kinematically based on the dynamics of hull and road arms as in Fig. 2. In the simulation terrain is modeled with a tremendous number of triangular polygons, which is shown in Fig. 3. The contact points between wheels and polygons must be determined to compute ground reaction forces. In order to locate the 14 wheels on polygons in large terrain database the bisection search was applied with a dynamic window for each wheel. The entire data base is too broad to be

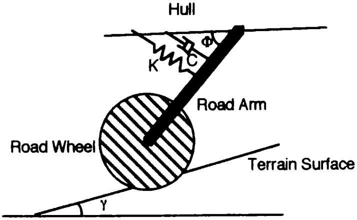


Figure 3. Geometry of wheel, road arm, hull, and terrain surface

searched in limited time. Taking advantage of hull and road arm dynamics, we can find the vicinities of the wheels' contact points. Candidate polygons are obtained by understanding the tank dynamics, which are called dynamics windows.

This idea can be extended to general dynamic systems such as aircraft, automobile, and other vehicles. For example, stability derivatives and engine parameters in flight simulation are typically stored in tables. They are functions of state variables, or related to state variables at least. Then dynamic window search can be applied by understanding the behaviors of state variables.

CONCLUSION

The study indicates that simple bisection search is superior to more sophisticated searching methods available in the field. A dynamic-window algorithm is also introduced in this investigation. If the data table is related to state variables of the dynamic system, a window, portion of the table, can be generated where the search key resides by considering the system dynamics. The window becomes smaller if the prediction of data dynamics is more accurate. It is another way for fast search, which is very practical for real-time use.

REFERENCES

1. Hoffman J., Numerical Methods for Engineers and Scientists, McGraw-Hill, 1992.

2. Gonnet G., George J. and Rogers L., "An Algorithmic and Complexity Analysis of Interpolation Search", Acta Informatica 13, pp. 39-52, Springer-Verlag 1980.

3. Burton F. and Lewis G., "A Robust Variation of Interpolation Search", Information Processing Letters, Vol. 10, pp. 198-201, July 1980.

4. Lewis G., Boynton N. and Burton F., " Expected Complexity of Fast search with Uniformly Distributed Data", Information Processing Letters, Vol. 13, pp. 4-7, Oct. 1981.

5. Franklin W., "Padded Lists: Set Operations in Expected (log log N) Time", Information Processing Letters, Vol. 9, pp. 161-166, November 1979.

6. Lin K., The Use of Function Generation in the Real Time Simulation of Stiff Systems, Ph.D. Dissertation, The University of Michigan, pp. 104-127, 1990.

AIAA-96-3517-CP

THE NASA 747-400 FLIGHT SIMULATOR: A NATIONAL RESOURCE FOR AVIATION SAFETY RESEARCH

Barry T. Sullivan, NASA-Ames Research Center *
Paul A. Soukup, NSI Technology Services Corp. **

ABSTRACT

This paper describes the NASA Ames Research Center's Current Technology Glass Cockpit Flight Training Simulator located at Moffett Field, California. This unique simulator is used to conduct aviation human factors and airspace operations research. The simulator is an exact replica of a cockpit of one of the most sophisticated and advanced airplanes flying in the world today. Although the simulator replicates a typical flight training simulator, it has unique research capabilities above and beyond the normal training simulator that is used to train today's airline pilots. This paper will describe the NASA simulator and it's advanced features, including it's unique research capabilities. It will also describe some of the research that has been conducted in the cab since its installation, and will also review some of the upgrades that are currently in progress, or will be conducted in the not too distant future.

INTRODUCTION

During recent years commercial airplanes have transitioned from cockpits equipped with analog instruments, gauges and switches to highly automated computer driven glass cockpits. Despite numerous advances in technology, most aviation accidents are still the result of human error. In order for scientists to study these types of errors a unique facility was needed which would enable them to examine the types of issues that are existent in today's modern cockpits. In late 1993, a unique Current Technology Glass Cockpit Flight Simulator was installed at the NASA Ames Research Center's Crew-Vehicle Systems Research Facility (CVSRF). This simulator provides researcher's and scientists with a means to explore issues pertaining to aviation human factors and airspace operations in a full-mission simulation environment. This unique simulator is but one integral component of the CVSRF. The CVSRF, located at Moffett Field in California, is a national research facility which was built in the early 1980's to study aviation safety issues. The CVSRF is comprised of two full-mission flight simulators - an Advanced Concepts Flight Simulator (ACFS) and a Current Technology Glass Cockpit Flight Simulator, as well as an air traffic control (ATC) simulator. The ACFS represents a "generic" operational flight deck of the future, while the Current Technology Glass Cockpit Flight Simulator reflects a state of the art aircraft equipped with today's latest technological advances. Each of the simulators are integrated with the facility's ATC simulation providing the capability to perform complex full-mission human factors and airspace operations studies. This paper will focus strictly on the features and capabilities of the Current Technology Glass Cockpit Flight Simulator and how these features are utilized in supporting national research programs related to improving aviation safety.

The NASA Current Technology Glass Cockpit Flight Simulator is an exact replica of a United Airlines Boeing 747-400 airplane cockpit. It represents a modern commercial transport aircraft incorporating highly advanced autoflight and guidance systems, including a sophisticated integrated avionics and warning system (A picture of the 747-400 simulator cockpit is shown in Figure 1). All systems within the simulator function and operate just as they do in the actual airplane, allowing researchers and scientists to conduct studies examining the human-to-human and human-to-machine interfaces encountered in the flight deck environment with a high degree of fidelity and realism. Unlike the typical flight training simulator used by the airlines, the NASA 747-400 simulator is enhanced with special research capabilities to support the many programs utilizing the cab. Customers originate from within NASA, industry, the airlines, universities and other government agencies such as the Federal Aviation Administration (FAA). Research objectives include developing fundamental analytic expressions of the functional performance characteristics of aircraft flight crews; formulating the principles and design criteria for aviation environments of the future; evaluating the integration of new subsystems in contemporary flight and air traffic control scenarios; and the development of new training and simulation technologies that are required by the continued technical evolution of flight systems and of the operational environment.

* NASA 747-400 Simulator Manager, AIAA Member
** 747-400 Simulator Project Engineer

SIMULATOR DESCRIPTION

The NASA 747-400 Simulator was built by CAE Electronics Ltd. and was installed at the CVSRF in September 1993. The 747-400 simulator is configured to United Airlines Tail #RT612 and is certified to the Federal Aviation Administration's (FAA) Level D requirements in accordance with *Federal Aviation Regulation (FAR) Part 121 Appendix H* and *Advisory Circular AC120-40B*, dated 29 July 1991. The simulator is also certified to the International Civil Aviation Organization's (ICAO) Level II International Qualification Standards, as defined in the international advisory circular, *International Standards for the Qualification of Airplane Flight Simulators*, dated January 1, 1992. These levels of certification are recognized by the FAA and industry as the highest possible levels of certification for airplane simulators in the world and provide immense credibility to the numerous research programs that are conducted in the NASA 747-400 simulator. The simulator compartment includes the cockpit flight deck, observer seats, an experimenter operator control station, and an engineering terminal. A floor mounted access stairway is provided to allow access to and from the flight deck compartment. Simulator features include a VITAL VIIe visual system, a digital control loading and motion system, a weather radar system simulation, and a traffic alert and collision and avoidance system. Other key features which will be described include the 747-400 cab's advanced aircraft avionics, integrated display system, digital sound/aural cues system, the simulator's hardware and software architecture, and it's unique research specific capabilities.

VITAL VIIe VISUAL SYSTEM

In compliance with FAA Level D requirements, the 747 cab is equipped with a Flight Safety International VITAL VIIe visual system. The VITAL VIIe with photo texturing and superior scene quality depicts out the window scenes in either day, night, dawn or dusk conditions. The visual system is a 3 channel, 4 monitor cathode ray tube (CRT) based system driven by a Motorola Delta series computer providing a 36 degrees vertical by 88 degrees horizontal field of view, using zero gap optics. The CRT monitors are raster-calligraphic and are capable of producing approximately 500,000 pixels and between 600 to 750 textured polygons per channel, as well as 1,000 calligraphic lightpoints for a day scene, and 5,000 lightpoints for night/dusk/dawn scenes. Numerous customized airport visual database scenes are simulated including airport scenes for San Francisco, Atlanta, Los Angeles, Boston, Chicago, Denver, Dallas-Fort Worth, New York, Honolulu, Hong Kong, Melbourne and Heathrow to name a few. Other customized databases can be developed through the use of an off-line visual modeling station. The ability to model terrain profiles is also possible. In addition, a generic airport scene capability is also provided in order to simulate non-customized airport scenes. Special weather scenes including weather fronts, thunderstorms, rain, lightning, hail, snow, fog and patchy fog are also available. Control of visibility, runway, airport and ambient terrain lighting, as well the presentation of other ground and air traffic is also available.

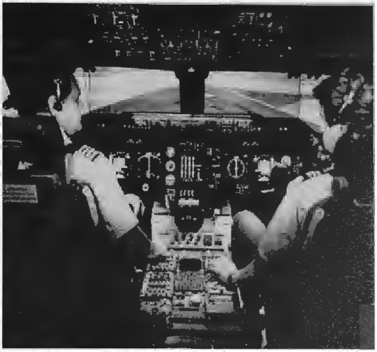


Figure 1 - Picture of 747-400 Simulator Cockpit

CONTROL LOADING and MOTION SYSTEM

Also in compliance with Level D requirements, the simulator includes an advanced digital control loading and six degree-of-freedom motion system. This system utilizes hydrostatic actuators powered by a remotely located hydraulic power supply to provide accurate motion and control feel forces. The hydraulic system consists of a CAE 600 series motion hydraulic supply unit consisting of two motor pump units connected in parallel. The hydraulic supply unit, connected to the motion base frame hydraulic system by flexible hoses, feeds the six servoactuator assemblies and the control loading actuators. The motion base frame hydraulic assembly consists of one pressure line, one return line, two drain lines for upper and lower servojack drains, and three precharged accumulators so that the system pressure does not fall below safe operational levels during worst case conditions. Hydraulic pressure required for control loading is supplied by the control loading pump in the hydraulic power supply unit. In the event of a control loading pump failure, the motion pump is also capable of supplying the control loading system with no adverse effect on motion performance. The cabinet housing the digital control electronics also serves as an interface to the host computer driving the simulator. This stand-alone DN1 cabinet houses the digital electronics, associated controls, display panels, power supplies, and test features required to control the motion and control loading systems. A CAE customized C30 board is the primary computing element in the DN1 based motion and control loading systems. The C30 computes aircraft manufacturer control surface models at a rate greater than 500 Hz. Control loading and motion servo loops run at an iteration rate of greater than 2 kHz. The software for the C30 is downloaded from the simulator host computer at load time. The C30 software communicates with the host-resident control loading and motion simulation through a serial ethernet link.

The motion system software provides cockpit movements in six degrees of freedom with movement in space of the cockpit of the simulated aircraft. The cockpit location is calculated as a moment arm from the aircraft's dynamic center of gravity. An adaptive filter software driver program is used to generate motion commands from linear and angular accelerations from the flight equations. The motion system provides physical sensations felt at the onset of an acceleration, which is then followed by low-level acceleration washout. Sustained longitudinal and lateral acceleration cues are created by use of the earth's gravity vector, and pitch and roll tilts of the cockpit. Buffet characteristics occur during appropriate flight conditions and match aircraft flight test data within very tight tolerances as determined in the *FAA Advisory Circular AC120-40B*. Special effects include the simulation of body accelerations, flap and gear buffets, stall buffets, high speed buffets, spoiler buffets, thrust reversers, engine vibrations, ground reaction forces, and weather effects such as turbulence, thunderstorms, and windshears.

The CAE 600 series motion system has an actuator stroke length of 54 inches and is capable of supporting a load of 22,000 lbs. It can simulate accelerations of +/- 1 g in the vertical direction and +/- 0.7 g's in the lateral and longitudinal axes. The maximum excursion is at least -37.5 deg to +32.5 deg in the pitch axis, +/- 32 deg in the roll axis, and +/-37.5 deg in the yaw axis. The maximum velocities are at least +/-30 deg/sec in the pitch axis and +/-32 deg/sec in the roll and yaw axes. The maximum accelerations are at least +/-250 deg/sec/sec in all three rotational axes.

The simulator also includes automated testing and tuning features to assure control loading and motion system fidelity. In addition, the simulator is man-rated to ensure that the simulator is safe and that all possible precautions have been implemented for all users and participating subject pilots using the cab. A series of interlocks have been implemented to ensure operational safety. In order for the motion system to be activated each of the interlocks must be closed. Should any of the interlocks be opened, the motion system will deactivate and come to a rest position, thus ensuring operational safety. Figure 2 depicts the 747 simulator on motion.

Figure 2 - Picture of 747 simulator on motion.

WEATHER RADAR SYSTEM SIMULATION

The 747 simulator includes a weather radar system simulation which is based on using stored weather cell shapes, which are scanned under software control to present a realistic radar display of a weather front. These fronts must be closely coupled with the 747 simulator's visual and motion systems to ensure simulation fidelity and realism and include precipitation and turbulence effects. Weather radar display information is presented to the pilots via the navigation system displays. Each of the weather cells has a vertical profile enabling the simulation of antenna tilt effects by displaying the projected image of the cloud horizontal plane at the intersection of the antenna boresight and the centerline of the weather cell. Special effects such as visibility, lightning and thunder can also be selected for added realism. Range attenuation and weather cell occultation are also simulated. Sensitivity time control and path attenuation compensation are also simulated with respect to receiver characteristics. Up to 20 weather fronts can be built and stored in the NASA 747's weather radar system simulation. Each weather front can contain up to 20 weather cells, with a maximum of 5 cells along any azimuth. Nonadditive mixing of weather cells is also possible to create new shapes by overlapping existing shapes. The weather radar system software resides partly in the host computer and partly in the weather radar processor board. The host resident software performs database management, radar control monitoring and data processing. The board resident software performs the weather cell scanning and the radar beam processing for occultation, range attenuation and compensation.

TCAS

The 747 simulator includes a Traffic Alert and Collision Avoidance System (TCAS II) which has been implemented according to the Minimal Operational Performance Standards (MOPS) RTCA DO-185. A TCAS II processor which provides the necessary interface to drive the 747 TCAS displays and the corresponding aural warnings for traffic and resolution advisories is simulated in the simulator's host computer. The host computer simulation software generates the flight paths for intruders and other traffic, as well as controls and formats all TCAS processor inputs and outputs. The 747's navigation display and primary flight display are used to depict traffic and resolution information. TCAS intruders can be controlled by one of two methods on the 747 simulator. The first is by the simulator's own TCAS control utility which can generate as many as 12 intruders at one time within a 15 mile radius. The second is by sending intruder information via the CVSRF's air traffic control simulator. Typical threats include traffic advisories, resolution advisories with no climb/no descend commands, resolution advisories with climb/descend commands, resolution advisories with further crew action required commands, and resolution advisories with further crew action required in the presence of a second threat commands. Transponder equipment for intruder aircraft can be selected by an operator as either mode A, mode C, mode S, or none. Intruder aircraft with mode A transponders provide only bearing and range information on the TCAS displays. Mode C and S transponders provide relative altitude, bearing and range information, and nontransponder equipped intruders do not generate any display information. Resolution advisories are provided for only mode C and mode S transponder equipped aircraft.

AIRCRAFT AVIONICS

The B747-400 simulator's advanced avionics includes seven aircraft Line Replaceable Units (LRUs) - two Flight Management Computers (FMCs), three Multi-function Control Display Units (MCDUs), an ARINC Communications, Addressing and Reporting System (ACARS) Management Unit, and a Ground Proximity Warning System (GPWS) Unit. The Honeywell FMCs handle flight performance management, navigation, guidance, thrust control, and display data processing. Through an ARINC interface, the FMCs receive information from other LRUs such as the MCDUs and the ACARS, and also from simulated systems such as the Inertial Reference System (IRS), Global Navigation Satellite Sensor Units (GNSSUs), radios, Air Data Computers (ADC), and fuel system. The data received by the FMCs must behave exactly as in the aircraft. Any discontinuity or irregularity will result in the FMCs acting abnormally. However, certain actions that do not occur in the aircraft cannot be avoided in the simulator. For example, maneuvers such as repositioning the simulation from point to point in a single iteration or freezing the simulation will cause undesired effects such as winds or inaccurate trajectory calculations. Software in the FMCs called SIMSOFT handles these situations by using an ARINC 610 protocol which will inhibit certain FMC inputs and computations. Outputs from the FMCs go to the host computer on the ARINC interface to be used as inputs to the various simulated systems such as the Integrated Display System (IDS), the Central Maintenance Computer (CMC), and the autoflight system. The FMCs also send outputs to the MCDUs, the ACARS, and the cockpit printer. The FMCs are loaded with operational software and a navigational database. The operational software also contains a performance database in order to provide the FMCs with data it needs to calculate pitch and thrust commands. The navigational database includes information that would be contained on navigational charts such as location of navigational aids, airports, runways and other selected airline information. New operational software can be loaded when required, and a new navigation database is normally updated every 28 days.

The three MCDUs, which are also built by Honeywell, allow the pilots to interact with the FMCs through a keypad and display. The majority of system messages and warnings are displayed through the IDS. However, the FMCs can generate forty-two different messages to be displayed in the scratch pad of the MCDUs. These messages are related to guidance and navigational items that require action by the flight crew. The two forward MCDUs permit the Captain and First Officer to interface with and monitor the FMCs, the navigation radios, the ACARS, and SATCOM. The center, rear MCDU only interacts with the ACARS, SATCOM, and CMC. SATCOM is a voice and data satellite based communications system that represents a significant improvement over HF radio for connectivity, voice quality, and traffic volume capacity, especially in the oceanic environment. The CMC software simulation is limited to warning system confidence tests and selection of various aircraft system maintenance pages.

The Collins built ACARS unit serves as a method of communication between a ground station and the B747-400 aircraft using VHF radio or SATCOM. The ACARS interfaces directly with the FMCs, the MCDUs, the VHF/SATCOM receiver, and the cockpit printer. ACARS has a voice and data mode, respectively. In the voice mode, conventional audio techniques are employed. While in the data mode, the flight crew transmits and receives data from a ground station via a digital data-link. The flight crew can send (downlink) position reports, fault reports for maintenance, and requests for information to the airline dispatcher. Information such as departure details, dispatch release, or free text can be sent (uplink) to the flight crew. In the simulator, information is data-linked via a dedicated interface card to the experimenter operator station (EOS). The simulated ground station on the EOS has the capability of receiving downlinks and generating uplinks.

The Sundstrom GPWS unit provides ground proximity warnings with appropriate voice, and other aural and visual indications.

DIGITAL SOUND/AURAL CUES SYSTEM

An 8-channel sound system with 19 speakers implemented throughout the entire flight compartment simulates the various sounds and aural warnings that are audible within the 747 flight simulator. Simulated sounds are generated, processed and controlled digitally using digital signal processing techniques. Attention to direction, frequency and amplitude of simulated sounds ensures realism. Controls have been provided for overall sound system volume, as well as software control to adjust the relative levels of simulated sounds. Simulated sounds are automatic and are representative of typical sounds heard throughout all phases of flight. These include aerodynamic hiss, engine surge, reverser sounds, compressor stall, runway rumble, explosive decompression, crash, rain, hail and thunder. The sound system produces the appropriate noises through a series of frequency generators, noise generators, and sound look-up tables. The sounds are derived from audio tapes supplied by the aircraft manufacturer. Data obtained from the spectral analysis of these recordings is entered into the host computer using various utilities. A serial ethernet and a Datapath C Microprocessor Controller DMC-16 card links the host computer to the sound cards in the sound chassis. Pure sounds are generated on a dedicated circuit card and sent to the appropriate loudspeakers via mixers and amplifiers. Other cards are dedicated to producing noise such as white noise and impact sounds. Using host based sound frequency analysis tools, sound measurements can be taken and verified against flight test results ensuring fidelity and realism. This is a requirement for FAA Level D certification.

The audio interface is composed of an audio chassis and a Digital Audio System Interface Unit (DASIU). In the audio chassis, signal generator cards produce various tones for communications and avionics including aural alerts generated as part of the Modular Avionics and Warning Electronics Assembly (MAWEA) system. These include bells indicating an engine or APU fire, unsafe warning sounds, caution sounds, selective calling, ground proximity warnings, altitude alerts, and stall warning alerts. A personal computer based digital voice system simulates the ATIS and VOICE.

SIMULATOR HARDWARE ARCHITECTURE

The simulator is composed of computers that simulate on board systems and stimulated computers that exist on the B747-400 aircraft. Taking advantage of IBM's latest technology over the last few years, the host computer driving the majority of the simulation, is a single processor IBM RISC 6000, Model 580. The host is networked to various computers including a separate data collection computer which is an IBM RISC 6000, Model 560, and four Silicon Graphics Inc. (SGI) Indigos which control the graphics for the two experimenter operator stations. Using a non-standard ethernet protocol for real-time and higher bandwidth, the host is connected to an interface that drives the simulator hardware. The simulation system interface includes a CAE standard input/output (I/O) interface, a special ARINC interface, a High Resolution Graphics Card (HRGC) interface, a control loading and motion system interface, a visual system computer interface, and an audio and sound system interface.

The CAE standard I/O interface handles analog signals for such systems as the flight instruments and discrete signals such as push buttons and circuit breakers. The ARINC-429 interface is a serial data bus that connects the host computer to the busses of stimulated aircraft components that utilize the ARINC-429 standard. The ARINC-429 protocol specifies a serial system for data interchange between different aircraft LRUs. The hardware portion of this interface is based on three types of boards that exchange and format data between the host computer via the ethernet and the ARINC DMC card.

The HRGC interface between the host computer IDS simulation and the six high resolution CRT displays in the cockpit consists of three chassis containing eleven High Resolution Graphics Cards and five DMC-16 boards. The HRGC is a microprocessor based card that has computing power comparable with an SGI IRIS workstation. It displays 1280x1024 pixels at a refresh rate of 60Hz. It is capable of simultaneous display of 256 colors and draw rates of 117,000 vectors per second and 44,000 filled triangles per second. The DMC-16 boards provides control and communications between the HRGCs and the host computer system. Eight HRGCs are used to drive the IDS pages and three are used for weather radar system applications. The HRGCs contain all the graphics information needed to drive the CRTs. A videoswitcher carries the red, green and blue (RGB) signals from one set of HRGCs to a given CRT. A block diagram depicting the 747

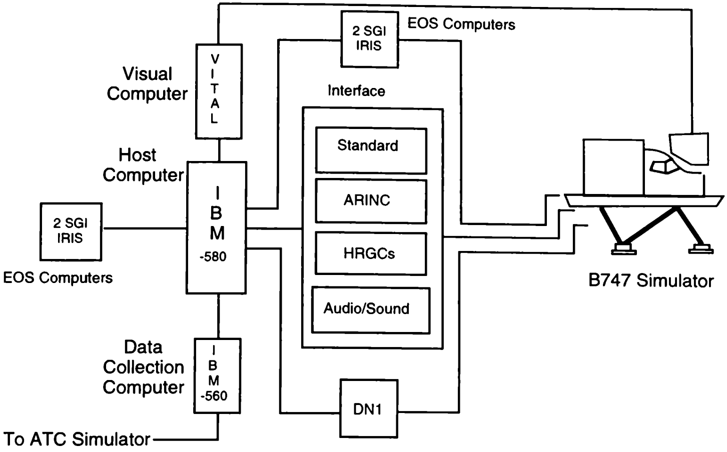


Figure 3 - NASA 747-400 Simulator Hardware Architecture

simulator's hardware architecture is shown in Figure 3.

SIMULATOR SOFTWARE ARCHITECTURE

All software on the IBM 6000s and SGI Indigos operate in a UNIX environment. The simulation software programs are written in high level languages such as FORTRAN or C, and are structured in a modular fashion. Configuration and control of the software is performed by a CAE developed real-time executive called SIMEX-Plus. It controls the linking, loading, executing, and scheduling of software modules. For configuration control the utility uses the load module concept. Each load module is a complete and independent package of the simulation called a configuration. Therefore, a configuration can be developed and maintained for an experiment without impacting other experiments. SIMEX-Plus also ensures that software resident on the simulator is properly controlled and that a history of changes is kept. Other utilities include a computerized test system which enables users to monitor, set, or plot global variables. The performance evaluation utility monitors the timing of all modules and processes so that users can be sure that no excessive computation is occurring. Also, a navigational database development tool allows users to create or modify navigational features such as radio aids or runways. There are also utilities for software maintenance and simulator status reporting.

All critical simulation models such as flight controls, aerodynamics, motion, flight instruments, aircraft position, and visuals execute at the critical rate of 30 Hz. The scheduling is dictated by two dispatcher programs, one for synchronous operations and the other for asynchronous operations. Modules are called at defined intervals anywhere from 33 to 266 milliseconds on the synchronous dispatcher, and from 50 to 300 milliseconds on the asynchronous dispatcher. A module's position in the dispatcher depends on the nature and significance of it's computations. Precise and balanced scheduling on the IBM RISC 6000-580 ensures that the facility enjoys a fifty percent spare time capability that can be used for development and other external purposes.

RESEARCH SPECIFIC CAPABILITIES

The NASA B747-400 simulator is unique in that its purpose is to support human factors and airspace operations research rather than being dedicated to flight crew training. This requires differences primarily in three areas - the ability to modify the flight displays and other flight crew interfaces, enhanced control over external effects, and the capability for powerful but flexible data collection and performance measurement.

PROGRAMMABLE FLIGHT DISPLAYS

The 747-400 is equipped with an Integrated Display System (IDS) which is composed of the Electronic Flight Instrument System (EFIS) and the Engine Indication and Crew Alerting System (EICAS). The EFIS portion of the IDS is composed of the Primary Flight Display (PFD) and the Navigation Display (ND) for both the Captain and First Officer. The PFD displays attitude/direction/altitude/airspeed information, while navigation, performance, and weather radar information is displayed on the ND. The EICAS displays engine data, caution and warning messages, and subsystem data and faults. On the B747-400 simulator, the IDS is programmable to support the development of new or revised flight displays and/or system synoptics to support aviation safety research. The actual aircraft IDS is normally comprised of six integrated display units (IDUs), which are driven by three EFIS/EICAS Interface Units (EIUs). The information displayed on the IDUs is controlled by the unique IDU location, control panel command, and automatic or manual source switching. On the simulator, the IDUs are simulated through the use of six commercial, raster-based CRT displays which are driven by the HRGC chassis. When the simulator is loaded, the HRGCs contain down load files which include all the EFIS and EICAS display information. The down load files can be modified in order to create or alter IDS display pages. The files can be edited using a CAE developed utility called TIGERS on a remote graphics workstation. The TIGERS utility is a windows based tool that allows the user to create or modify dynamic objects with attached attributes such as color, scale, or position that can be driven from the host computer. The IDS simulation model, resident on the host computer, provides the EFIS and EICAS system logic to drive, in real-time, the HRGC graphics software on the IDS pages. The software also provides for EIU functions, including data processing and formatting for EICAS message generation, and interfacing with other simulator hardware and software systems. Using the videoswitcher, there is no limitation as to which IDS page can be displayed on which IDU, giving the user even greater flexibility. The IDS simulation facilitates system modifications on an experimental basis without being dependent on aircraft hardware. The NASA 747 simulator was the first U.S. simulator with programmable flight displays to receive FAA Level D certification.

EXPERIMENTER OPERATOR SYSTEM

The Experimenter Operator System on the NASA B747-400 simulator, not unlike training simulators, provides the capability to initiate, monitor, and control activities in the cockpit during a simulator session. However, the NASA 747 flight simulator has additional features such as an off-board experimenter operator station (EOS), repeater monitors, data collection facilities, and the ability to connect with the CVSRF ATC simulator and other simulators outside the facility. Each EOS is composed of two touch-screen SGI IRIS display systems providing both display of current information and user friendly, touch screen controls for manipulating the simulator. In addition, both the on-board and off-board stations provide control of the simulator through programmable control panels (PCP's) and various buttons and switches. Each EOS enables the operator to manipulate or control a variety of variables via different pages and windows, (i.e. the position set page establishes the simulator on the runway or gate, or trimmed in flight at any airport in the world that can support a B747-400 aircraft). Weather is altered on the weather set page where winds, temperatures, turbulence, and special effects such as lightning, thunder, or blowing snow can be set. Aircraft weight and balance are controlled on the aircraft configuration page. Approximately three-hundred system malfunctions are accessible through individual malfunction control pages. However, the primary means by which an operator can control an experiment and the flight tasks in a repeatable manner are through scenarios developed off-line in advance of the experiment, using a PC-based Scenario Editor Utility (SEDU). Scenarios provide the researchers with the ability to automate and control configurations and events during a simulation by preprogramming a series of EOS functions. Another important tool available on the EOS is the capability of creating a "snapshot" of the simulator that can be recalled at a later time. Snapshots can be permanently saved as "setups" and then used within a scenario. This feature is important in the research environment so that a researcher can repeat a procedure from a precise initial starting point in order to attain reproducible results. An additional benefit to the researchers are the off-board EOS station repeater monitors which display all six cockpit displays and an out-of the-cockpit scene to allow the researchers to observe, in real-time, exactly what the pilots are seeing inside the cab.

EXPERIMENT DATA COLLECTION

The B747-400 simulator is equipped with a specialized system for collecting run data during a simulator session. A list of variables and the rate at which they will be captured is created on the IBM RISC 6000-560 data collection computer before the session. The name of the list is then entered on the EOS to initiate the recording of the data. The information is recorded by the data collection computer and is stored on hard drives. An average of 2000 parameters (4 bytes) at 30 Hz can be recorded on contiguous disk space of up to 11.3 gigabytes. Experimental data can also be collected in the form of audio and video. The EOS can control the on/off state and vary the volume of up to eight different microphones in the cockpit or at the EOS stations. The audio channels are then mixed together via a digital/audio mixer to provide an analog output to the off-board speakers and the video recorder. In addition, six miniature, low light video cameras are installed around the cockpit giving the researcher the ability to videotape six specific views plus an out-of-the-cockpit view during the simulator session. Time-stamped video/audio recording combined with digital data collection gives the researcher a powerful tool to study the crew and crew-to-vehicle interactions during a simulator session or experiment.

AIR TRAFFIC CONTROL (ATC) SIMULATION INTERFACE

The CVSRF has a separate air traffic control (ATC) simulator which provides realistic air-ground communications and coordinates the appearance of aircraft visible from the simulator cockpit. This system is hosted on a separate computer, a VAX-6310. The ATC simulation consists of four air traffic controller stations and four "pseudo-pilot" stations. Each one of these stations is equipped with multi-channel voice disguisers and are linked to each other, as well as to the 747 simulator. The four "pseudo pilot" stations are able to control numerous pseudo-aircraft at one time. The ATC simulator interacts with the communications, TCAS, and visual systems on the B747-400 host computer. The EOS provides a control switch which determines whether the pseudo aircraft are displayed as TCAS intruders on the B747-400 or whether the intruders originate from the host computer's preprogrammed TCAS scenarios. This gives researchers the option to monitor and record the interaction between the flight crew and ATC, or the flight crew with researcher-designed TCAS scenarios. The ATC simulator is connected to the host computer via the data collection computer through a direct memory access (DMA) link. This link consists of a DR11W interface residing on the data collection computer and a DRB32W interface on the ATC simulator's VAX 6310.

LINKS TO OTHER SIMULATION FACILITIES

The B747-400 simulator is currently linked with two other facilities: the FAA Technical Center in Atlantic City, and the FAA Aeronautical Center in Oklahoma City. The FAA Technical Center includes a unique air traffic control simulation complex which is used to conduct airspace operations research. The facility is linked to several other simulators throughout the country via a high speed digital voice/data communications system. The 747 is linked to the Technical Center by two discrete voice and data lines respectively, which operate at 9600 bps each. These four lines are fed through a multiplexor that combines all of the voice and data communications into a single 56 Kbps dedicated line. The 747 is one of several participating simulator's around the country, each of whose lines are combined into a single 1.544 Mbps T-1 line at the FAA Technical Center. The voice lines transmit all of the flight crew-controller communications. The data lines transmit flight simulator state information such as airspeed, position and altitude to the Technical Center and ATC information such as the time and aircraft designation to the flight simulator.

An additional link to the FAA's Aeronautical Center in Oklahoma City allows FAA personnel to monitor real-time results of an on-going experiment. This link employs a simple modem protocol over the telephone lines. Flight simulator state information such as position, airspeed, altitude, spatial deviations, and auto pilot modes are sent at 9600 bps.

RESEARCH PROGRAMS

Although the NASA 747-400 flight simulator is maintained to the highest recognized certification standards for training simulators, it is not used for training. It is used strictly for aviation human factors and airspace operations research. Participating subject crews are line-qualified 747-400 pilots, thus further validating the value of the results of the research programs conducted in the simulator. Since the simulator's activation in the fall of 1993, over twenty studies have been conducted in the NASA 747 simulator. Typical studies include the following:

Data-Link: A series of studies have been conducted by NASA and FAA to examine the use of digitized information transfer for the presentation of air traffic control information. These studies are concerned with trying to reduce the number of incidents due to communications problems between pilots and air traffic controllers. The results of these studies will help in the future design and development of guidelines and procedures for the implementation of digital information transfer in future commercial transport aircraft.

Converging Approaches: The FAA's System Capacity Office developed a program to focus on the improved utilization of instrument landing system (ILS) converging approaches and examined the related operational aspects towards improving efficiency. Currently, ILS terminal procedures (TERPS) missed approach primary obstacle clearance surface criteria impose capacity constraints on ILS converging approach operations. The FAA investigated procedures utilizing flight management system equipped aircraft to conduct simultaneous converging missed approaches from decision heights between 500 to 700 feet above ground level. Results from these studies showed that converging approaches were possible and could be performed safely using lateral offsets from the localizer between 90 to 100 degrees, thereby allowing decision height minimums to be reduced. Implementation of these new procedures are currently in process for Chicago O'Hare and Dallas-Fort Worth Airports using a reduced decision height of 650 feet.

Center TRACON Automation System Descent Advisories: This experiment examined the compatibility of a new air traffic control Center TRACON Automation System (CTAS) descent advisory clearance with the operational procedures available on current commercial aircraft. The intent of this study was to identify and eliminate any problems with the delivery, timing or execution of successfully flying CTAS descent advisories. CTAS is a new air traffic controller automation tool which helps controllers schedule traffic more efficiently. Results of this study will help develop new phraseology and guidelines for using CTAS descent advisories which will eventually be tested during field tests at Denver International Airport.

Moving Map Display: This NASA study evaluated the use of an integrated moving map display for presenting ground taxi information which enabled pilots to navigate about the terminal area in low visibility weather conditions. Navigation data to the map display was provided via a simulated differential global positioning system (DGPS). The information was presented to participating flights crews on their navigation displays via data-link. The enhanced map display depicted a topographical view of the terminal area including runways, taxiways, gates, position of other aircraft, ATC clearance information, relative aircraft position, heading and ground speed. Results of this study indicated a definite reduction in crew workload in using the integrated map display versus the use of the conventional paper Jeppesen maps that are typically used today. In addition, crew's taxi time performance was improved as well by using the enhanced map display. Other potential benefits possibly gained by this type of display include improved terminal area capacity by enabling airplanes equipped with this advanced technology to operate in reduced visibility conditions, and a possible increase in safety by reducing the chances of possible ground incursions.

Multiple Parallel Approaches Program: This on-going FAA program evaluated the traffic handling capabilities of conducting multiple simultaneous independent instrument landing system approaches in instrument meteorological conditions. These studies evaluated the feasibility of conducting dual and triple parallel runway operations with varying runway separation distances by taking advantage of advanced radar systems such as the Precision Runway Monitor (PRM). This advanced radar system employs a faster radar sweep, allowing more accurate tracking of terminal area traffic. The 747-400 simulator was one of several simulators connected to the FAA Technical Center's air traffic control simulation complex via a high speed digital voice/data communications system. The results of this study provided the FAA with a new separation standard for parallel approach operations.

3-D Audio: This NASA study evaluated the use of spatialized sound techniques for detecting other ground traffic during taxi operations in the terminal area. For this study, directionalized auditory warnings were briefly enunciated over customized (stereo) pilot headsets to attempt increasing the crew's situational awareness about non-visible hazards such as other nearby aircraft, or as annunciations for intersecting taxiways during taxi operations. The main thrusts of this study were to determine the time to complete taxi routes under spatially-audio assisted and non-assisted conditions, incursion warnings and taxiway announcements. Preliminary results indicated that all participating pilots expressed a strong preference for the ground collision avoidance alert to be included with a future system, hopefully decreasing the amount of time a pilot would need to respond to a possible ground incursion.

Propulsion Controlled Aircraft: This study evaluated the use of a propulsion only flight control system in the event in which an airplane's primary flight control system malfunctioned or became inoperative. This study made use of control laws developed at NASA Dryden to control aircraft flightpath angle by manipulating aircraft thrust to maintain pitch and roll control during approach and landing operations. Results of this study indicated that participating pilots were very excited about the fly-by-throttle concept. A follow-on study is currently being planned that will extend the use of the propulsion controlled flight algorithms to include engine out performance as well as other phases of flight for a four engine aircraft.

These programs are just a representative sample of the type of programs that are supported on the NASA 747-400 simulator. Over the upcoming years, the 747 simulator will be used extensively in support of NASA's Terminal Area Productivity (TAP) and Advanced Air Traffic Technologies (AATT) Programs. In addition, NASA will continue to work with the FAA in trying to resolve human factors and airspace operations issues by supporting the FAA's National Plan for Human Factors and Free Flight Programs.

FUTURE PLANS

Although the NASA 747-400 simulator represents a state of the art aircraft, it too will evolve over the upcoming years. Probably the most significant upgrade to the simulator which is currently taking place in the airplane as well, is the implementation of the Future Air Navigation System (FANS). FANS is an advanced avionics system upgrade that will utilize global satellite based information for communications, navigation, surveillance and air traffic management for the twenty-first century. The FANS upgrade is primarily a Boeing-Honeywell implementation to upgrade the 747-400 with FANS compliant avionics, taking advantage of satellite navigation and communications systems, and technology advances in automation. FANS will utilize Global Positioning System (GPS) information for aircraft tracking and navigation, supporting enroute and terminal area non-precision approaches. In the future, ground based augmentation is expected to extend the GPS capabilities by including precision approaches. Other FANS modifications to the simulator include upgrades to the various installed avionics, specifically the two FMC's, the three MCDU's, the Multi-input Cockpit Printer, the ACARS Management Unit and SATCOM. Also, there are changes integrated as part of the FANS package which require modifications to the appropriate simulated systems including the 747-400's EICAS, MAWEA system for pending data-link messages, and the modification of the navigation displays depicting the use of GPS data for primary navigation. New key features resulting from the FANS upgrade include Automatic Dependent Surveillance (ADS) allowing more precise aircraft tracking, Airline Operational Communications Data-Link (AOC DL) of flight plan information, winds forecast data and route modifications, and Air Traffic Control Data-Link (ATC DL) of air-ground messages including clearance uplinks. In the real world environment the ADS and ATC DL functions are generally hosted on controller workstations with graphical user interfaces for selecting or displaying the data-link information for the airplanes flying in the airspace in the controller's jurisdiction. For the simulated environment, the interaction for ground support functions such as AOC DL, ATC DL or ADS are provided via specially designed control pages on the 747-400 simulator's Experimenter Operator's Station (EOS). All data-link applications have airborne and ground-based counterparts which exchange data through specially formatted messages and are transmitted via an ACARS network. The airborne side of these applications are included as part of the upgraded avionics. The ground based applications required development for a simulated environment. Some of these features include Required Time of Arrival (RTA) utilizing time based navigation, and Required Navigation Performance (RNP) which compares actual position versus required position on a given route. The significance of the FANS upgrade is that it will enable the CVSRF to support important national programs such as NASA's AATT Program and the FAA's Dynamic Aircraft Route Planning (DARP) Program. These two programs will rely very heavily on the new capabilities provided by the FANS upgrade.

Other upgrades envisioned over the upcoming years include the integration of some advanced display symbology depicting potential conflict alerting schemes, enhanced collision and avoidance logic, reduced separation, vertical situation, and advanced ground taxi displays to name a few. These advanced displays will take advantage of the 747 simulator's reprogrammable flight display capabilities, and will attempt to increase aviation safety by hopefully reducing pilot workload. In addition, integration of a heads up display is being considered for the 747-400 simulator. Technology advances in the future will enable the integration of a smaller and less obtrusive HUD system which can be used to study advanced HUD symbologies that will allow pilots to navigate about the terminal in low or zero visibility conditions. As an extension of the FANS upgrade, integration of ADS-B is envisioned to be installed in all aircraft some time in the not so distant future. This will allow aircraft to determine the detection and flight path of other aircraft in the nearby vicinity representing an enhanced traffic and collision avoidance capability.

SUMMARY

The NASA 747-400 Flight Simulator is an essential and vital tool for studying issues pertaining to aviation human factors and airspace operations research. Since its inception, numerous studies have been conducted aimed at improving aviation safety and the overall efficiency of the national airspace system. This unique facility has enabled and will continue to enable scientists to develop and test new concepts in a realistic cockpit environment through the use of full-mission simulation. The NASA 747 simulator will allow researchers to conduct studies that examine how crew members interact with each other and how they interface with advanced automation concepts in the ever changing flight deck environment. The facility will continue to make an impact on the current and future aviation environment by it's continuous support of important national programs such as NASA's TAP Program, AATT Program, the FAA's National Plan for Human Factors and Free Flight. This facility will continue to play a vital role in enhancing aviation safety over the years to come.

REFERENCES

1. CAE Boeing 747-400 Full Flight Simulator for NASA, Volume 1: Technical Specification, TPD 08593-1 Rev. 3, CAE Electronics Ltd., St. Laurent, Canada, June 1992.

2. Shiner, R., Sullivan, B., Man-Vehicle Systems Research Facility: Design and Operating Characteristics. AIAA/AHS Flight Simulation Technologies Conference, Hilton Head Island, South Carolina, August 1992.

3. CAE Boeing 747-400 Full Flight Simulator, Software Documentation, Software Avionics, Integrated System Display, 10015347 Rev. 0, CAE Electronics Ltd. St. Laurent, Canada, January 1994.

4. CAE Boeing 747-400 Full Flight Simulator, Audio System Maintenance Manual, TPD 10483 Rev. 0, CAE Electronics Ltd., St. Laurent, Canada, January 1994

5. CAE Boeing 747-400 Full Flight Simulator, Serial Communication System Maintenance Manual, TPD 10485 Rev. 0, CAE Electronics Ltd. St. Laurent, Canada, March 1994.

6. CAE Boeing 747-400 Full Flight Simulator, Sound System Maintenance Manual, TPD 10488 Rev. 0, CAE Electronics Ltd., St. Laurent, Canada, October 1993.

AIAA-96-3518-CP

THE NASA ADVANCED CONCEPTS FLIGHT SIMULATOR: A UNIQUE TRANSPORT AIRCRAFT RESEARCH ENVIRONMENT

Matthew W. Blake, NASA Ames Research Center

ABSTRACT

This paper addresses the current and planned capabilities of the NASA Ames Research Center Advanced Concepts Flight Simulator (ACFS) located at Moffett Field, California. The ACFS is used to study many aspects of human factors in aviation safety as well as methods to improve aviation operational efficiency. The ACFS provides full mission functionality and appears similar to high fidelity training simulators but is unique in that no flight hardware is used; the entire simulation is programmable. This capability allows the simulator to be reconfigured to meet any research requirement. This paper describes the current and planned capabilities of the ACFS, elaborates on some of the unique features available to the aviation research community, and describes some of the specific research programs completed recently as well as some future research plans.

INTRODUCTION

Modern commercial transport aircraft are extremely complex systems. In addition to the conventional control stick or yoke, rudder pedals, and thrust lever, the modern flight deck is comprised of numerous computer displays, keyboards, and cursor control devices. Rather than describing a flight as "flying from point A to point B", the task of piloting a commercial aircraft from one location to another is now more aptly referred to as "managing the flight".

The current Air Traffic Control (ATC) environment is also very complex. The airspace is rigorously controlled and there are very complex procedures and rules regulating operation of aircraft in the airspace. Utilizing current procedures, the ATC environment is near capacity. New ground based automated ATC systems hope to significantly increase capacity, and rapid changes in satellite navigation and on-board computational capability have provided aircraft with the ability to determine more efficient routes than the current ATC system. These possible changes to the airspace system all require thorough development and test prior to the fielding of a new system. Due to this complexity of the aircraft and air traffic systems, the pilots often have so much flight planning and system management work to accomplish that they let the on-board computers actually guide and control the aircraft during much of the flight. Even when the pilots choose to interactively guide the aircraft, they will often use a combination of autopilot knobs and switches to provide a heading and altitude rather than the conventional stick and rudder approach to control of the aircraft. This use of automation can improve the efficiency of the operation tremendously, however it presents many new problems. These include pilot or ATC controller confusion on what the system is doing, why the system did it, and what the system is going to do next. These changes have brought around a whole new set of potential human-machine interface hazards to the safe execution of commercial transport flight operations.

To accurately study human performance in the airspace operations arena or on the flight deck requires providing the full complexity of the entire environment. To safely study this requires use of a simulation rather than experimentation in the real world. The ability to simulate this complete system is called full-mission simulation. In the early 1980s, NASA scientists recognized the need to perform research in this area and the Crew-Vehicle Systems Research Facility (CVSRF) was constructed at the NASA Ames Research Center. The facility contains three main components; a conventional current technology aircraft simulator, an advanced future technology aircraft simulator, and an air traffic control simulator. The facility has gone through numerous upgrades and changes since this initial capability leading to the current configuration.

The current CVSRF conventional technology aircraft is a Boeing 747-400 simulator. The 747-400 utilizes computer driven all glass display instrumentation and the latest in avionics systems. This is an exact replica of a current state of the art aircraft. The simulator adheres to FAA Level D certification standards, the highest certification standards available. In addition to all conventional 747-400 aircraft features, the simulator includes several capabilities not available on a training simulator. These include programmable flight displays and extensive data collection capabilities. The 747-400 simulator is an excellent platform for studying proposed near term airspace operations changes and human factors issues related to incremental system changes.

The ACFS has evolved over the years and is currently being completely rebuilt. Recent upgrades to the ACFS include a new multi-processor host computer, a new instrument panel including flight displays and computers, an updated digital communication network, a new aural cue system, and a new cockpit center pedestal and throttle system. The new ACFS includes a complete full-mission simulation of a mid size, mid range transport. Unlike the 747-400 simulator that utilizes aircraft avionics units, the entire ACFS is simulated on general purpose computers and is therefore completely re-programmable. Any system can be changed to address specific research areas. The ACFS includes a relatively conventional baseline flight deck environment that can be easily reconfigured for any research design. The Flight Management System (FMS) is also fully programmable, providing a unique research capability. The ACFS is an excellent platform for studying significant changes to the airspace system or major new concepts in human interfaces.

The ATC environment is a significant contributor to pilot workload and, therefore, to the performance of crews in flight. Full mission simulation is greatly affected by the realism with which the ATC environment is modeled. The CVSRC ATC simulator provides the capability to integrate the aircraft simulators into a very complex airspace environment. This environment includes several ATC sectors, dozens of pseudo aircraft, and an elaborate communication and data-link system to simulate the large traffic load experienced in the real world. The ATC simulator can also tie in with other simulators and the NASA Ames developed Center/Tracon Automation System (CTAS). This system is designed improve the safety and efficiency of traffic flow in the terminal area.

The ACFS and 747-400 simulators can be configured to operate together and with the ATC simulator. Additionally, both the CVSRF simulators can be linked to outside simulation facilities.

SIMULATOR DESCRIPTION

The ACFS was originally developed as a joint effort between NASA Ames, NASA Langley, and the Lockheed Georgia Company. Three simulators were built and located at each site. This paper describes the NASA Ames simulator.

The ACFS is currently configured to simulate a generic advanced twin engine mid-range transport aircraft, similar to a Boeing 757. The ACFS consists of a re-configurable cab on a 6 degree-of-freedom motion platform with a Link Image II Out The Window (OTW) visual system. The cockpit includes a conventional center console with throttle levers and aircraft system control and display devices, an overhead panel for control of many other aircraft subsystems, and, following the current upgrade, 8 video display screens across the main panel for flight, guidance, navigation, and status information. There is a Mode Control Panel (MCP) in the glare shield for autoflight control and 2 Control Display Units (CDU) for interfacing with Flight Management Computers (FMC). A time lapse photograph of the simulator in motion is shown in figure 1.

Figure 1: Time lapse photograph of ACFS in motion

AIRCRAFT PERFORMANCE

The ACFS performance and behavior can be tuned to a particular experiment's requirements. The default configuration is a low wing airplane with twin turbofan engines. General operating characteristics include:

- Maximum gross weight - 224,000 lbs.
- Payload 60,000 pounds; capacity - 200 passengers
- Twin engine - 41,000 pounds rated thrust per engine
- Speed - 0.78 Mach; range - 2500 miles
- Fuel capacity - 42500 lbs. usable

AIRCRAFT SYSTEMS

The ACFS aircraft represents a combination of current state of the art systems and some improved systems. New or modified systems can be incorporated easily to meet research objectives. The basic features of the current existing simulated aircraft systems are described here.

The flight control system consists of conventional flight surfaces (rudder, ailerons, elevators, flaps, slats, and spoilers) employed in both normal and innovative ways (continuous adjustment of ailerons to reduce wing bending, adjustment of flaps and ailerons to reduce drag, and the use of spoilers coupled with the elevator to maintain glidepath). Wing tip devices are also employed to alleviate twisting forces. Manual and automatic trim of aileron, stabilizer, and rudder are available. Flight control is purely fly-by-wire. The Stability Augmentation System (SAS) is active in the pitch, roll, and yaw axes.

The engines are advanced technology high bypass turbofans with electronic engine control. Features include full function from startup through shutdown including fire detection and extinguishing. The fuel system includes full authority digital fuel control, integral fuel heating system, boost pump, APU fuel control, isolation and fuel shuttoff valves. Engine status is displayed on a compact engine display similar to the Boeing 757.

The Auto Flight System (AFS) consists of the Autopilot Flight Director System (AFDS) and the Autothrottle System (A/T). When engaged, the Flight Management Computer (FMC) automatically manages pitch, roll, and thrust through simultaneous control of the AFDS and A/T. Control of the AFDS is accomplished through the MCP. The AFS functions very similarly to an advanced Boeing commercial transport.

The Flight Management Computer (FMC) is one of the most advanced programmable FMCs available anywhere in the research community. The system includes accurate vertical and lateral flight planning, navigation, and guidance. There is currently a major development effort underway to increase the simulated FMC capability.

Warning systems include an Engine Indication and Crew Alerting System (EICAS), a stall warning system, a Ground Proximity Warning System (GPWS), and a Traffic Collision Avoidance System (TCAS).

Communication and Navigation systems include dual Very High Frequency (VHF) transceivers, dual VHF Omnidirectional Range (VOR)/ Distance Measuring Equipment(DME) receivers, dual Instrument Landing Systems (ILS), and an Inertial Reference System (IRS) consisting of three Inertial Reference Units (IRU). Frequency control of receivers is through a digital entry panel and includes active and standby capability.

Electrical system components include four engine driven generators, an auxiliary power unit (APU) with two generators, external power capability, emergency circuit breakers, air-conditioning, heating, avionics, and associated controls and displays. Power is distributed over six separate buses and can be re-routed in several ways.

Aircraft exterior lighting includes landing, taxi, strobe, navigation, and anti-collision lights. Interior lighting control includes panel, overhead, and instrument intensity. The environmental systems include cabin pressure and temperature and emergency oxygen and de-pressurization control. Adverse weather systems include engine ice detection, anti-ice, continuous engine ignition; and pitot, Angle of attack sensor, and windshield heat. The landing gear system includes manual gear extension/retraction, anti-skid braking with carbon composite brakes, and a parking brake system.

FLIGHT DECK

The ACFS flight deck is designed for operation by a two-pilot crew. The instrument panel can easily be re-configured to a particular experiment's requirements. The default configuration is:

- McFadden hydraulic control loading sidearm controllers for pitch and roll control
- McFadden hydraulic control loading rudder pedals
- Back-driven throttle levers with reverse thrust, very similar to a Boeing 757
- MCP on the glare shield for autoflight control, very similar to a Boeing 757
- Two CDUs in the center pedestal for interfacing with the FMC
- Mechanical switches/knobs in the overhead panels and the center console for miscellaneous aircraft system control

The physical location of the flight displays in the instrument panel can be altered to meet research requirements. What is displayed on each display can also be reprogrammed to meet research requirements. The display installation conforms to ARINC specifications.

The default flight display configuration is:

- 8 inch Primary Flight Displays (Pilot and Co-pilot), similar to Boeing 777
- 8 inch Navigation Flight Displays (Pilot and Co-pilot), similar to Boeing 777
- 8 inch Engine/EICAS display in the centerline
- Additional 8 inch display below Engine/EICAS for researcher use
- Two 6x7 inch displays for system status and control for Pilot and Co-pilot
- Touch screens and cursor controls for interaction with systems displayed on 6x7 inch device

Figure 2 shows the flight deck as used until recently. Figure 3 shows a diagram of the flight deck that is currently under construction and is described here.

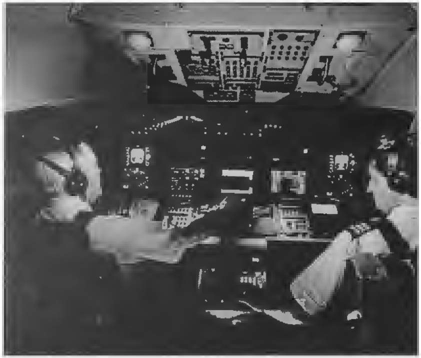


Figure 2: ACFS flight deck prior to overhaul

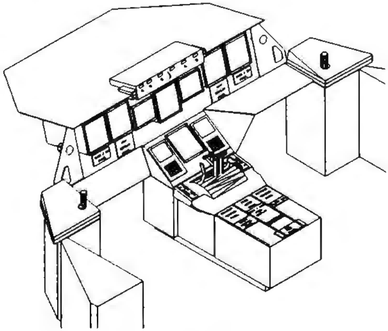


Figure 3: New ACFS flight deck design

SIMULATION CUEING SYSTEMS

In addition to the flight deck instrumentation and control devices, to pilots are cued by a high fidelity visual, aural, and motion cueing system.

The Out-the-Window (OTW) visual image is generated by a Link-Miles Image II visual system. The Image II visual system is a compact flight simulator attachment which presents computer-generated color scenes representing the outside world. These scenes depict specific airports and their surroundings as viewed at dusk, twilight, or night from the cockpit. Enroute visual scenes are also simulated as are other aircraft.

The aural cues are provided by a new Advanced Simulation Technology Inc. (ASTi) sound system. This system generates general aircraft sounds (engines, wind, landing gear in transit), aircraft warning sounds (GPWS alerts, EICAS alerts), and communication/navigation sounds (VOR identifiers, Automated Terminal Area Service (ATIS) messages).

The motion cues are provided by a Link Flight Simulation hydraulic hexapod motion platform. This system is capable of providing motion cues in all six axes. The motion of the crew station in the simulated aircraft is calculated in the host computer and that motion is filtered to create a smaller version that can be reproduced with the limited travel of the simulator motion system. The translational and rotational motion is converted into leg movements and sent to the motion system. The motion system is capable of providing acceleration and velocity cues in excess of:

• Vertical	0.8g acceleration, 24 inches/sec
• Lateral	0.6g acceleration, 24 inches/sec
• Longitudinal	0.6g acceleration, 24 inches/sec
• Pitch	60 deg/sec^2, 20 deg/sec
• Roll	60 deg/sec^2, 20 deg/sec
• Yaw	60 deg/sec^2, 20 deg/sec

SIMULATOR ARCHETECTURE

The simulation is distributed over many subsystems. A majority of the simulated systems are running on the host computer, a Silicon Graphics Inc. (SGI) Challenge multi-processor machine. The flight displays are all running on separate SGI single processor computers. There are separate systems for sound generation, OTW visual image, cockpit control signals, and other miscellaneous devices. Figure 4 shows a high level diagram of the major subsystems in the simulator.

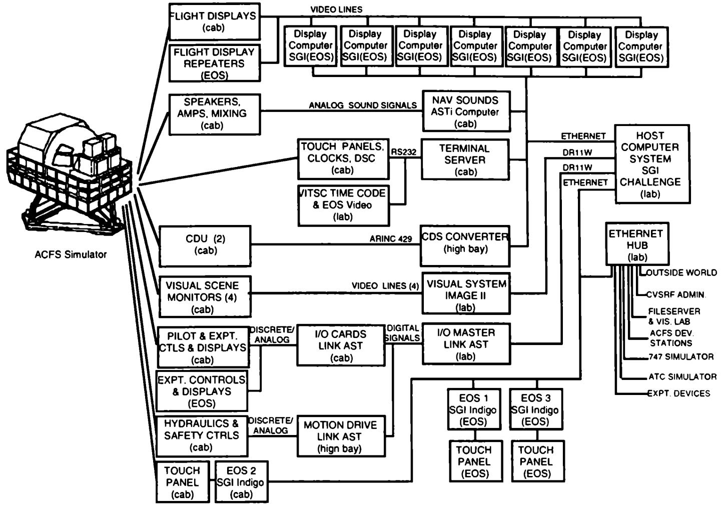


Figure 4: High level diagram of the major ACFS subsystems

Within the multi-processor host computer the software is separated into functional units and run on separate processors. The general load is configured as follows:

Processor-0: UNIX Operating System, Experiment Operator Stations (EOS), Asynchronous Input/Output (I/O), background processes
Processor-1: Main aircraft simulation (Controls, Aerodynamics, Equations of Motion, etc.) synchronous I/O
Processor-2: Data Collection
Processor-3: FMC process

The host computer is capable of utilizing 12 processors. We are currently using only 4. The majority of the data communication between computers is accomplished utilizing socket protocols over Ethernet lines. The ACFS software includes approximately 500,000 lines of code, split approximately evenly between FORTRAN and C.

EXPERIMENT OPERATOR STATION

Control and monitor of the active simulation is done utilizing a separate process running on the host computer. This process is called the Experiment Operator Station (EOS). This is not a hardware station but rather a program that presents buttons, menus, and diagrams on a color computer display for the engineer or operator to use for control or monitoring of simulation parameters. Many copies of the EOS may be run concurrently, allowing different engineers to monitor different parameters and view different pages of information. There are pages for general simulation control, aircraft configuration, weather, data collection, etc. Generally, additional new pages are created for each experiment's unique requirements.

In the main simulation control room/experimenter station there are also repeater displays of the crew's flight displays as well as mechanical switches and indicators for all safety related simulator controls. There is also an audio station allowing experimenters to communicate with the flight crew during an experiment or with observers located "on-board." Communicating with the ATC simulator is also possible from the experimenter stations. The EOS computer processes are run on two workstations located in the control room. These workstations have touch screen overlays in addition to traditional keyboard and mouse input devices.

There is also an EOS station located in the simulator cockpit. This location has an SGI workstation for running the EOS process as well as simulation control and communication hardware.

EXPERIMENT DATA COLLECTION

Digital data collection of experiment parameters is performed on a separate processor on the host computer. The system is capable of capturing data at multiple frequencies up to 30 Hz. Data is collected on a hard disk and can be processed or transmitted immediately following termination of an experiment run. The maximum recording capacity is 2000 words (8000 bytes) at 30 Hz., up to a maximum of 2 Gigabytes.

In addition to digital data, both audio and video analog data can also be collected. Low light video cameras are mounted in the simulator cockpit and are capable of recording pilot actions and instrumentation status. Audio recordings can be made of all audio channels and mixed with the video. In order to correlate the digital data with the audio and video analog data, a digital time code stamp is recorded and displayed on the video tape. Additional digital data from the simulation can be recorded on part of the video screen if desired.

The combination of audio and video correlated with the digital data provides a powerful tool for analyzing crew-vehicle performance.

SIMULATION DEVELOPMENT AND TEST

The entire SGI based simulation can be run on any SGI workstation in the facility. This "mini-ACFS" system provides an excellent initial development environment. When running on single processor workstations, the simulation cannot run in real-time but the individual processes stay in sequence so behavior for testing and debug is usually quite adequate. The engineer can run the existing flight displays and EOS as well as a graphical version of the mechanical MCP to provide an essentially complete development environment. Some workstations have been configured with two monitors to provide additional display space for this purpose. This mini-ACFS capability decreases the need for expensive duplicate simulators for development and significantly decreases the need for dedicated time on the main simulator early in the development phase.

The development computer supplier, SGI, provides a state of the art software development environment including a suite of Computer Aided Software Engineering (CASE) tools. This includes a Graphical User Interface (GUI) front end to the debugger, a static code analyzer, an on-line help utility, and much more. The key elements of this environment can be utilized with the real-time system so the user is not forced to develop and test his software utilizing different tools. This cuts down on development and test time and the user does not need to learn and stay proficient at two different environments. Also, the system support staff does not need to support a special (often in-house developed) real-time control, monitor, debug system.

RESEARCH PROGRAMS

The ACFS has been used for a broad spectrum of airspace operations and human factors research. Much of the early use was geared toward full mission studies of many concepts in cockpit design. Some examples include flight deck ergonomic studies on location and characteristics of the sidestick controls, flight display location and content, electronic checklists for coping with onboard malfunctions, and airport surface navigation maps on flight displays. More recent studies have looked at the implications on pilot workload of air-ground data-links and developed autoflight mode change prediction and intelligent cockpit procedural aid tools. The most recent studies are listed below.

Integrated Mode Management Interface (IMMI): This study evaluated an entirely new method of presenting autopilot mode status and control in an effort to improve the pilots awareness of current and projected autopilot actions. This study addressed what is termed "mode confusion" which is rapidly becoming recognized as a major contributing factor in many aviation incidents. The experiment showed the value of vertical situation displays and enhanced mode information.

Propulsion Controlled Aircraft (PCA): This study evaluated an emergency backup Fly-by-Throttle control system for use in the case of a complete conventional flight control system failure (no rudder, aileron, or elevator). The simulation successfully demonstrated tremendous improvement in the ability of the pilot to land safely over manual throttle controlled flight. This effort directly supported eventual flight tests on a McDonnell Douglas MD-11 transport aircraft at NASA Dryden Flight Research Center.

Neural Controlled Aircraft (NCA): This study utilized an active learning Neural Net model of the engine in an effort to further improve the PCA control capability, specifically during higher levels of turbulence and out of trim conditions. This system demonstrated the concept of utilizing active learning Neural Nets to automatically redefine the control system following an unidentified system failure. This experiment also demonstrated the power of a blended control system where all control surfaces and the engines are utilized in unison. For a damaged aircraft, this blended control system provided the pilot with much better performance than the standard flight control system which is based on control of independent axes.

3D Audio for TCAS: This set of experiments studied the use of 3D audio signals to provide directional cues to the pilots for Traffic Collision Avoidance System (TCAS) alerts in a heavily loaded full mission environment. Each crew flew from San Francisco to Los Angeles with standard TCAS monaural warnings and with the 3D Audio warnings. During the flight they were presented with several hundred other aircraft (traffic) and many of these became actual collision threats which activated the warning system. The experiment showed that crews could visually identify the threat following the warning message faster when provided with the 3D spatial cue.

The unique products of these experiments are directly beneficial in commercial aviation and private industry. The general public will see the results of such experiments by the improvement in aviation safety and efficiency. In the next few years, the ACFS will be used extensively in support of NASA's Terminal Area Productivity (TAP) Program and Advanced Air Transportation Technology (AATT) Program. Additional studies will be performed in support of other human factors research and FAA research programs. Some specific planned experiments include use of Head Up Displays (HUD) for improved Low-Visibility Landing and Surface Operations (LVLASO), Free Flight Collision Avoidance, and Free Flight/Flight Plan Negotiation.

FUTURE PLANS

As a research facility, the simulator is essentially always undergoing some form of renovation. The current upgrades were more extensive than is normally done for a specific experiment. These modifications will be completed by the end of August, 1996. In the next year, the simulator will go through the final planned upgrade which includes replacement of the current OTW visual system and cockpit I/O system. A Head Up Display (HUD) system will be installed to support specific TAP research programs.

Other major improvements include a continuing effort to improve the ACFS simulator's software FMC to meet the AATT research objectives. The simulation will be required to support large data-link capabilities in both the ground ATC side and the airborne FMC side. Additional capability for features such as Required Time of Arrival (RTA) and Required Navigation Performance (RNP) will be implemented.

SUMMARY

The ACFS provides full-mission aircraft functionality including elaborate autoflight and flight management systems yet the hardware and software configuration can be changed as needed to address specific research requirements. Unlike most commercial aircraft and training simulators, the ACFS does not include any actual aircraft avionics boxes so the entire simulation is fully programmable to address any research requirements. The unique research capabilities of the ACFS enable scientists to develop and test new concepts in a realistic cockpit environment through the use of full mission simulation. Experimental research performed on the ACFS directly supports many NASA programs in airspace operations and human factors. The ACFS will continue to make an impact supporting basic research as well as focused areas such as the Terminal Area Productivity (TAP) program and Advanced Air Transportation Technology (AATT) program.

References

1. Shiner, R., Sullivan, B., Man-Vehicle Systems Research Facility: Design and Operating Characteristics. AIAA/AHS Flight Simulation Technologies Conference, Hilton Head Island, South Carolina, August 1992.

2. Sexton, G., Crew Systems and Flight Station Concepts for a 1995 Transport Aircraft, NASA Contractor Report 166068, April 1983.

3. Aircraft Operations Manual for the Advanced Concepts Flight Simulator (ACFS), AP-1015, NSI Technology Services Corp., February 1996.

AIAA-96-3519-CP

UTILIZATION OF A GROUND TAXI MAP TO SUPPORT LOW VISIBILITY SURFACE OPERATIONS

Vernol Battiste, NASA-Ames Research Center
Michael Downs, Sterling Software
Barry T. Sullivan, NASA-Ames Research Center, NASA 747-400 Simulator Manager, AIAA Member
Paul A. Soukup, NSI Technology Services Corp., 747-400 Simulator Project Engineer

Abstract

This paper will report the results of a simulation which examined the effects of various kinds of information instantiated on different maps on pilot's ability to taxi at Chicago-O'Hare under low visibility conditions. Three groups of pilots landed and taxied a FAA level D certified Boeing 747 simulator in three visibility conditions during a series of 24 landing-to-gate arrival sequences. In one condition, only existing navigation aids, such as a Jeppesen paper map and an ATC clearance, were provided. The second condition added a moving map of the airport which depicted the current position of the aircraft on a highly detailed map. The final condition added graphical ATC assigned route and traffic overlaid on the map with digital (text) datalink ATC clearance in a text window on the display. Measurements of taxi speed, route following accuracy, and workload were compared. Video and audio data recordings of flight deck activity also allowed us to assess user interaction with the system. In addition, user input on the display design was assessed using questionnaires and post-simulation interviews. Comparisons of taxi performance, workload and user feedback showed that crews in the ANDS group performed ground taxi navigation significantly better than crews in the paper or basic map groups. Crew workload was reduced with either electronic map compared to the map information currently being used by flight crews (paper maps). Finally, crews responded favorably to the ANDS display design and were able to utilize the system to support the ground taxi navigation task.

Introduction

The efficient movement of airport traffic under all weather conditions is critical to improving the efficiency of gate-to-gate movement of aircraft in the National Airspace System. In any vehicle, operators must relate map position data with either direct visual information or sensor data presented on displays to determine or confirm their current location, answering the question: "Am I where I think I am?, or where am I?" As weather conditions deteriorate and visibility is reduced, the information needed to answer these questions may not be available, which can reduce the efficiency of ground operations.

The key to improving the safety and efficiency under these conditions is cockpit aids that improve navigational and situational awareness.

The primary objective of the Low Visibility Landing and Surface Operations (LVLASO) element of the Terminal Area Productivity program (TAP) is to improve throughput and to increase capacity during low visibility conditions at high density airports. The goal of the surface operations element of the this program is to improve ground operational efficiency and to enable operations during low visibility conditions.

Thus, a goal of the map element of TAP is to provide information to the crew which will enable the safe and efficient movement of aircraft during low visibility conditions. Based on interviews with pilots during operations in visual meteorological conditions (VMC) and instrument meteorological conditions (IMC), research on navigation and geographical awareness, and display design research, a prototype advanced navigation display concept was design (1) (2) (3) (4). The map display design effort focused on developing an integrated navigation display system which would provided information similar to the information available in a clear weather visual scene, to support operations down to CAT IIIb weather conditions. The features of the display were: (1) aircraft's current position, taxiway center lines, other traffic, and ATC clearance information, including any restrictions, graphically depicted on an airport layout (map); and (2) integrate vehicle control, navigation, and operational situation information in an intuitive and quickly identifiable and understandable manner. An objective of this integrated design, based on the proximity principle, was to reduce pilot's head down time. [5] To support the ground taxi task requirements and user needs the display was designed to be: (1) *easily interpretable;* (2) *dynamic* - - selectable north-up or track-up map orientation, 2 sec. update rate, selectable map scale and display size; (3) *responsive* - - operator configurable; (4)

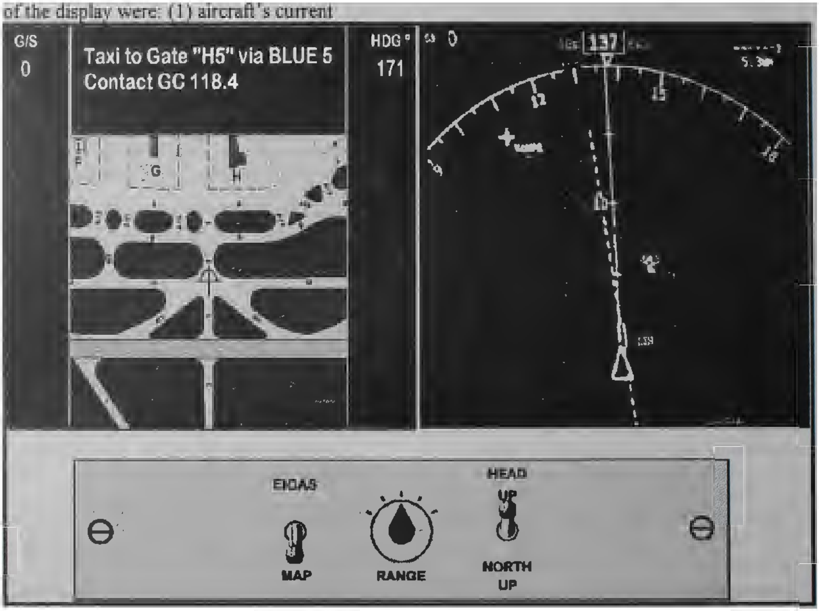

Taxi to Gate "H5" via BLUE 5
Contact GC 118.4

Figure 1: Advanced Navigation Display, B-747 Navigation Display, and Flight Deck Control Panel.

highly accurate -- map database to support ground navigation with DGPS data; and (5) *integrated* - assigned runway, taxi route, and assigned gate; heading, and ground speed, and other similar groupings on a single display.

The display concept was installed in the NASA CVSRF Boeing 747-400 Flight Simulator where the map was presented on each pilot's navigation display, see Figure 1. The study compared ground taxi navigation performance of 747 line flight crews with three map configurations during simulated landing and taxi operations at Chicago O'Hare airport under three visibility conditions. After concluding the experimental trials the crews completed two questionnaires and participated in a post-simulation design discussion. The questionnaires and discussions were designed to allow crews to participate in the design of the display concept.

Methods

Subjects: 12 current Boeing 747 Captains and First Officers participated in the study. All crews (12 captains and first officers) who participated in the study were Boeing 747 flight qualified flight crews. With one exception all crews were line flight crews; the one exception being a training crew from a major airframe manufacturer. The median flight time for our flight crews was 1800 hours for Captains and 1500 hours for First officers. The median flight simulator experience was 300 hours for Captains and 237 hours for First Officers.

Simulator: The baseline NASA 747-400 Simulator is an exact replica of a United Airlines Boeing 747-400 airplane cockpit. It represents a modern commercial transport aircraft incorporating highly advanced autoflight and guidance systems, including a sophisticated integrated avionics and warning system. All systems within the simulator function and operate just as they do in the actual airplane, allowing researchers and scientists to conduct studies examining the human-to-human and human-to-machine interfaces encountered in the flight deck environment with a high degree of fidelity and realism. Unlike the typical flight training simulator used by the airlines, the NASA 747-400 Simulator is enhanced with special research capabilities to support the many programs utilizing the cab, making it an ideal platform to conduct studies dealing with aviation human factors and airspace operations issues. This unique simulator is but one integral component of the Crew-Vehicle Systems Research Facility (CVSRF), located at NASA Ames Research Center. Another important component of the CVSRF is an Air Traffic Control (ATC) simulator, which can be integrated to the 747-400 Simulator, providing the air-ground communications that pilots are normally exposed to, creating a "full-mission" environment with the ability to perform complex multi-aircraft scenarios to support aviation safety research.

The NASA 747-400 Simulator was built by CAE Electronics Ltd. and is configured to United Airlines Tail #RT612. It is certified to the Federal Aviation Administration's (FAA) Level D requirements. The simulator is also certified to the International Civil Aviation Organization's (ICAO) Level II International Qualification Standards. These levels of certification are recognized by the FAA/ICAO and industry as the highest possible levels of certification for airplane simulators and provide immense credibility to the numerous research programs.

The simulator includes: a VITAL VIIe visual system, a digital control loading and motion system, and advanced aircraft avionics including a highly automated autoflight and guidance system. The simulation is hosted on an IBM RISC 6000, Model 580 computer. The cockpit displays are programmable to support the development of new or revised displays. In addition, cameras and audio speakers are mounted throughout the cockpit to allow videotaping of pilots actions or images of the various flight displays or Multi-function Control Display Units. A separate IBM RISC 6000 Model 560 is also provided for data collection and reduction purposes, allowing researchers to collect as much as 11 gigabytes of data during an experiment session.

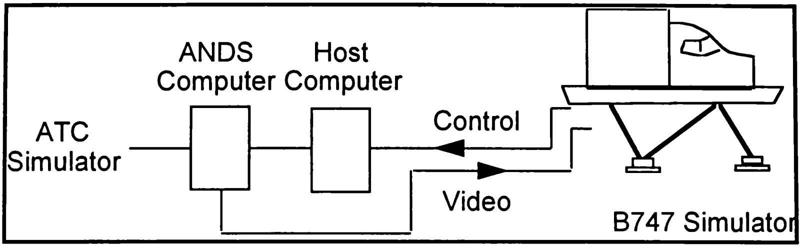


Figure 2: ANDS simulation control network.

User-friendly, touch-sensitive experimenter operator control stations are also provided, as well as repeater monitors of each of the cockpit displays, including an out-the-window visual scene. These capabilities provide researchers with powerful tools to study flight crews and crew-to-vehicle interactions during an experiment.

Simulator modifications: Numerous modifications were made to the 747-400 Simulator to support this study. The implementation of the ground taxi map in a commercial aircraft would primarily require differential global positioning system (DGPS) information of the own aircraft and other traffic, control inputs from the cockpit, and display enhancements for the ground map display. These requirements were met with minimal modifications to the simulator.

Simulated DGPS positional data of the own aircraft was derived from the simulator's navigation model. External traffic position information was taken from the CVSRF's ATC simulator where the traffic data was generated. The traffic consisted of up to nine aircraft which were broadcasted to the host computer every 500 milliseconds. All positional data was placed in packets which were comprised of an aircraft identification number, latitude, longitude, altitude, heading, speed, and time. The position packets were sent to the ANDS computer from the simulator's host computer at a rate of twice a second.

The only hardware cockpit modification required to support this experiment was the installation of a customized control panel just above the throttle quadrant, see Figure 1. The control panel allowed flight crews to switch from the ND to the map, adjust the map scale, and change the map orientation between north-up and present heading-up. The control positions were made available to the simulator's host computer with modifications to the hardware interface. The scale and alignment mode inputs were placed in a packet and transmitted to the ANDS computer only when there was a state change.

The control and position packets were transmitted to the ANDS computer via an ethernet link using TCP/IP and sockets protocol , see Figure 2. Two other packets were also developed: (1) to inform the ANDS system of the start and finish of an experiment run, (2) the disconnect packet, which was sent before the simulator dropped the sockets connection.

Using the control panel, flight crews could display the ground taxi map in lieu of their Navigation Displays (ND), see Figure 1. On the NASA B-747 Simulator, the cockpit flight displays are simulated through the use of commercial, raster-based cathode ray tubes (CRT). The CRTs require a red, green, and blue (RGB) signal with horizontal and vertical synchronization on green. The simulator uses a software controlled, video switching device to determine which cockpit display is presented on which CRT. The video switcher has the capacity for depicting the six cockpit displays, one blank to emulate a malfunction, and a spare. Utilizing the spare channel in the video switcher, the ground taxi map could be presented on any of the six cockpit CRTs. Position information from the simulator's host computer was utilized by the ANDS computer to generate the position of ownship and traffic on the map display. The ANDS computer then transmitted a 1 volt RGB signal which produced the 10 inch by 10 inch (1280x1280 pixels) ground taxi map image on the ND.

Another requirement for this experiment was the development of an extremely accurate visual scene of Chicago's O'Hare Airport. This scene developed from scratch, depicted each of the active runways used for this study, including ramps, gates, terminal buildings, taxi-ways, including centerlines, and taxi-way signs. In order to get a high fidelity and accurate airport model, airport maps were obtained and then digitized to portray the airport scenes. Several iterations of alignment were conducted between the visual scene and the ground taxi map display to ensure that each of these scenes transmitted the same positional information. This was critical when displaying other aircraft and their relative positions on both the ground taxi map display and in the out-the-window scenes during ground operations. In addition, the visual scenes and ground taxi map display also had to be aligned with the CVSRF's ATC simulation.

Experimental Conditions: A 3 X 3 mixed factorial design was employed to assess crew ground taxi performance. Twelve flight crews were assigned to three experimental groups: paper maps, paper map plus basic electronic map, and paper map plus advanced navigation display system.

The Jeppesen KORD Airport map of O'Hare International was the paper map used in the study.[7] Additionally, crews were provided coded taxi routes, similar to the O'Hare Jeppesen route, used for departures to facilitate clearance delivery in all map conditions. The basic moving map provided ownship position associated with a highly accurate and detailed depiction of the airport layout, coupled with current speed and heading (see Figure 1) . The Advanced Navigation Display System, in addition, provided a data linked Air Traffic Control Assigned routing in both a text and as a graphic depiction overlaid on the map, and traffic (position of other aircraft operating on the airport), (see Figure 1).

Crews performed normal landing and ground taxi operations at Chicago O'Hare airport under three visibility conditions: VFR, 600 Runway Visual Range (CATIIIB), and 300 RVR. Twenty four taxi routes of equivalent length and number of turns were presented in a randomized order to each crew.

After completion of the 24th experimental run all crews were exposed to the advanced map display design during one low visibility taxi operation, and asked to provide feedback on the design during a post-simulation design review.

Data collection: Crew taxi time from the runway to the ramp, navigation errors, TLX workload ratings, video/audio tape, and questionnaire data were collected during the study.

Procedure: On Day 1 of the study each crew was briefed on the goals and objectives of the study. The crews were instructed to follow normal company procedures for flight and ground operations. The crews were also briefed on the scenario, which started at 12 mile final on course and glide path with clearance for the approach on tower frequency.

In the basic and ANDS map conditions the crews were allowed to switch from the NAV display to the Map at their discretion. After landing and clearing the runway crews received a coded taxi clearance from the ground controller. They were instructed to follow the clearance in a safe and efficient manner to the clearance limit (their arrival gate). Upon arrival at the ramp they were instructed to taxi clear of the main taxi way and to park and set the brakes, which concluded the automated data collection.

After each trial, individual crew members were instructed to complete a NASA TLX workload rating. Crews were briefed on the configuration of the 747-400 Simulator, and those in the moving map groups were briefed on the basic and ANDS map system. Finally, they were escorted to the cab by a CVSRF pilot and an experimenter and allowed to conduct three practice trials to familiarize themselves with the simulation environment and with the map display system (basic or ANDS). Crews in the basic and ANDS map groups were allowed to develop their own procedure for switching from the NAV display to the ground taxi map.

Approximately 18 experimental trials were completed on Day 1 of the study. On Day 2 the crews completed the remaining experimental trials. Crews in the basic moving map and paper map groups were presented one additional trial, where they were allowed to land and taxi with the ANDS map to facilitate the post-simulation design discussions. Prior to the post-simulation design discussion each crew member completed two questionnaires: (1) demographic and experience - which provided information about themselves and their flight experiences, and (2) Advanced Navigation Display Design - which allowed them to critique the map design. After completing the questionnaires, each crew provided feedback on ground taxi operations with and without the ANDS system.

Results and Discussion

During ground taxi operations in the Boeing 747 aircraft the Captain normally conducts all taxi operations. In this study Captains conducted 75% of the ground taxi operations, thus this paper will generally focus on the Captain's performance and workload data. However, where appropriate we will include crew and First Officer's data. When the Captain is taxiing the aircraft he focuses primarily on information in the visual scene to support navigation and geographical orientation. As the visibility is reduced, the availability of this information is also greatly reduced. Thus, he depends more on the First Officer to aid him in acquiring information from the visual scene, and to supplement the visual information with information from charts and other instruments (heading indicator). Also, since the amount of time a feature is available and the number of features in visual scene is reduced with the visibility, Captains tend to focus primarily on the external scene, while the First Officer focuses on information in the cockpit.

Performance

Taxi Time represents the total duration of the taxi segment, measured in seconds. Timing began when the nose of the aircraft crossed a pre determined line from the runway onto the active taxiway. Timing stopped when the nose crossed from the taxiway onto the ramp. This method of measuring taxi duration avoids any distinction between crews who terminated the run after crossing onto the ramp and those who taxied all the way to the gate before stopping.

Taxi time between the runway and the gate was significantly reduced when crews had either of the electronic maps vs. crews with the paper map, See Figure 3.

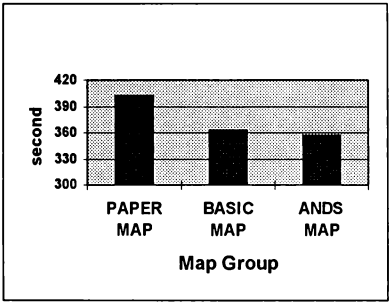


Figure 3: Taxi time from runway to ramp by taxi map condition.

Captains performing ground taxi navigation with either electronic map display were able to reduce their taxi time by at least 10% over Captains performing this same task using paper maps. During reduced visibility operations, the reduction in taxi time

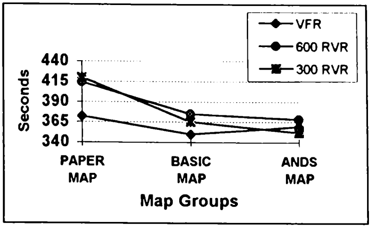


Figure 4: Taxi time by visibility by taxi map condition.

between the paper map and electronic map groups was even more dramatic - at 600 RVR a 10% decrease in taxi time; and at 300 RVR a 17% decrease in taxi time. Captains' average taxi time between the runway and the ramp, using paper maps in 300 RVR, was seven minutes (420 sec.), while Captains using the electronic map with route overlay (ANDS) averaged five minutes and fifty three seconds (353 sec.) see Figure 4.

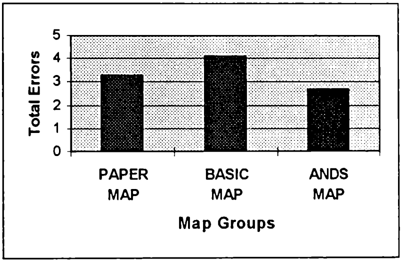


Figure 5: Total number of taxi navigation errors by map group

<u>Errors:</u> A number of factors were combined to account for the total number of navigational errors: (1) incorrect turn (at the correct location the crew turned in the incorrect direction; (2) missed turn (did not

initiate a turn at correct location); and (3) deviation greater than 20 feet from the taxiway centerline (early turn, late turn, and taxing into the grass). Crews in the ANDS map group made fewer errors overall (Figure 5), and were able to recover from an error significantly faster than crews in both the basic and paper map conditions, see Figure 6. Even though our flight crews were very experienced, both in the aircraft and in simulation, they had very little experience conducting ground taxi operations in a simulator (Captains 14 hours, and First Officers 12 hours). This factor may account for some of the maneuvering errors found during the study. However, the combination of faster taxi time and reduced errors, for the ANDS group, shows that their performance in terms of taxi time is not the results of a speed/accuracy tradeoff. Which seems to the true for the basic electronic map group.

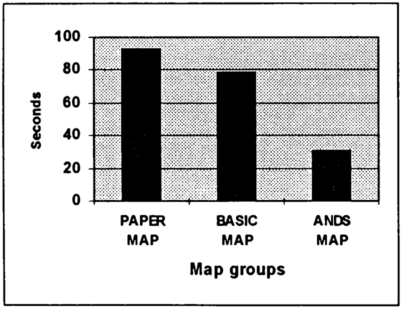


Figure 6: Mean recovery time by map condition.

Average (%) head-up time: This variable measures the total time that the crew member spent looking out the 'window', with his or her head up. It was measured as a percentage of the Taxi Duration. Captains' heads-up time was significantly reduced with increases in map information content. They averaged 84% head-up time with the paper map, and 65% head-up time with the electronic chart with route overlay, see Figure 7. Crew feedback during post-simulation interviews indicated that they focused primarily on the external scene in the paper map condition because of the amount of time needed to acquire information from a paper map.

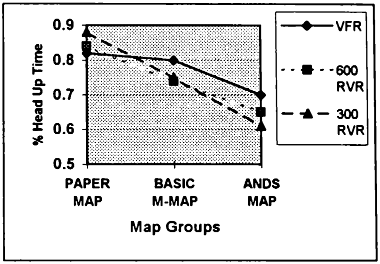


Figure 7: Captain's head up time by map group and visibility.

Average duration of a head-down response: Average heads down time represents the average amount of time that

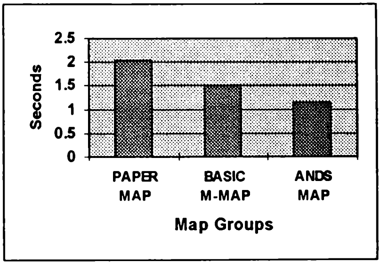


Figure 8: Captain's head down duration by map condition.

the crew member spent looking inside the cockpit before looking outside again. This variable demonstrates how quickly the crew member could obtain the information he or she was looking for. When Captains were taxiing, their head-down interval was reduced significantly with the electronic map and route overlay. This difference represent a 32% reduction in map assessment time, see Figure 8. The First Officer's head-down interval was also significantly reduced for the map with route overlay group, however the head-down interval increased with the basic map over the paper map group (mean paper 5.85 sec., basic map 6.69 sec., map with route 3.98 sec. This difference also represents a 32% reduction in head-down interval for First Officers in the map/route group. The additional time available to both the Captain and First Officer could be used to assess the rapidly changing information in the visual scene.

Frequency of head-down responses for both crew members was significantly increased with the map and route overlay. When the Captains were taxiing, their head down frequency was increased 260 % over Captains in the paper map group (mean head down frequency: paper map 28, basic map 39, and map/route 74). A similar result was found for First Officers, a paper map 34, basic map 30, and map/route 49). This result, combined with the shorter head down interval suggests that the integrated map display was used more efficiently to support navigation and situational awareness.

Workload

The NASA Task Load Index (TLX) was used to assess crew workload.[8] Captains in the paper map group rated their workload significantly higher than Captains in either electronic map groups, see Figure 9. Captains in the map with route overlay group rated their workload significantly higher than Captains in the basic map group. First Officers' workload ratings followed a similar pattern to Captains' (mean: paper map 28.74, basic map 18.52, and map/route 22.43). The workload difference between the two electronic map groups was not significant. However, both Captains and First Officers in the map with route group rated their workload higher than crews in the basic map group.

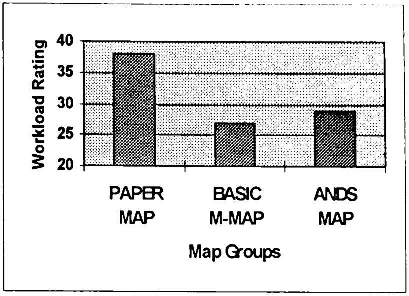


Figure 9: Captain's workload ratings by map conditions.

Questionnaire

Two questions from the Advanced Navigation Display study questionnaire, which was administered post-simulation, were selected to summarize crews' overall response to the display design.

1. Was the basic content of the map useful (basic airport layout: runways, taxiways, ramps, and other infrastructure)? All crews agreed that the basic information content of the map was useful, and on a 1 to 10 scale (1 not very useful, 10 very useful) Captains rated the basic layout 9.4, and First Officers rated the display 9.7.
2. Rate the display overall as a tool to support low-visibility navigation, on a zero to ten scale; zero not very useful, and ten very useful. Captains rated the display 9.4 and First Officers rated it 9.8.

Selected comments from Captains and First Officers:

- "More taxiway labels" are needed...
- "Could you rotate the text so it is always right side up?"
- "Need a planning mode to show airport while airborne"... "maybe on the lower EICAS display..."
- "Stop bars should be included on the display"
- "Inclusion of audio incursion alerts would be helpful"
- "Compass rose should be incorporated".. (Analog information to show trends not digital).
- "Position display could update more quickly" (We use a 2hz update rate in the study).
- "Enormously helpful, like to see some improvements in using display to begin turns. Maybe ... just some more training or experience needed"
- "The display was inherently clear and useful. Location on the airport was rarely in doubt".
- "I think its just wonderful the way it is. Great Job, great invention that major airlines need, to facilitate/save time and money for them, not to mention safety.
- "I like it. When can I have it in my airplane? Thanks."

Conclusions

Crews taxied significantly faster with the advanced map with route overlay. Their taxi time from the runway to the gate was reduced by 11% during 600 RVR, and 17% during 300 RVR operations, relative to the paper map condition, thus, suggesting increased benefits as visual obscuration increases. Even though crews using the map with route overlay had increased head down time, the duration of the head down response was reduced. The reduction in head down duration and the increased frequency of head down responses could allow crews to sample the rapidly changing visual scene more often, thus improving their overall situational awareness.

Crews experienced significantly lower workload while navigating with either electronic map display, although crew workload was higher with the map with data link ATC messages, route overlays, and traffic. Since this display provided additional information which could be used to more precisely position oneself relative to the ATC assigned route, and one's position relative to other traffic, this result is not surprising. Finally, crews had many comments and suggestions for improving the overall display design; some should be instituted immediately and others may not be possible.

Even though crews performed well with this system, many refinements are needed to overcome some persistent errors; such as over-steering and under-steering at route interchange points. This type of error was probably due to the visual disparity between aircraft's position on the map and line of site position on the airport. Also, the increased disparity in aircraft position with increased speed based on a 2hz DGPS update rate. Some of these errors could be easily ameliorated with head up guidance. Thus, we need to continue to pursue the integrated T-NASA system, which integrates a conformal head up, moving map and 3-D Auditory displays, to support ground taxi operations during reduced visibility conditions. Others could be eliminated by integrating technologies such as DGPS data with the Inertial Navigation System on the aircraft. The integrated system could provide continuous output to the map display, which would eliminate the speed/update problem.

References

1. Andre, A. D.,(1996). *Operational Issues During Low-visibility Taxi Operations: A Field Study*. NASA Ames Research Final Report Terminal Area Productivity Program. Cooperative Agreement NCC2-486.

2. Battiste, V. and Delzell, S., (1991). *Visual cues to geographical orientation during low-level flight*. In the proceedings of the Sixth International Conference on Aviation Psychology, pp. 566-571. Columbus: Ohio State University, Department of Aviation.

3. Wickens, C. D. and Battiste, V. (1994). *Cognitive Factors in Helicopter Low-Level Navigation: An Overview of the Research.* Presented at the American Helicopter Society 50th Annual Forum, Washington DC, May 11-13, 1994.

4. Battiste, V. and Downs, M., 1992. *Development of an Electronic Chart Display system (ECD) with Global Positioning Satellite (GPS) data*. Proceedings of IEEE/AIAA 11th Digital Avionics Conference, pp. 423- 427. Seattle: Washington.

5. Wickens, C.D. and Andre, A. D., (1988). *Proximity Compatibility and the Object Display*. Proceedings of the 32nd Annual Meeting of the Human Factors Society. Santa Monica, CA: Human Factors.

6. Sullivan, B., Soukup, P. *The NASA 747-400 Flight Simulator: A National Resource for Aviation Safety Research*. AIAA Flight Simulation Technologies Conference, San Diego, California, July 1996.

7. Sanderson, Jeppesen, (1995). *Coded Taxi Routes for Departure*. Jeppesen Airway Manual Services, Illinois.

8. Hart, S.G., Battiste, V., and Lester, P.T., (1984). *Popcorn: A supervisory Control Simulation for Workload and Performance Research*. In the Twentieth Annual Conference on Manual Control (pp. 431-454) Washington, DC: NASA Conference Publication 2341.

AIAA-96-3520-CP

HIGH-FIDELITY SIMULATION AS A LOW COST ENHANCEMENT TO FLIGHT TEST

Michael T. Brunner *
Flight Vehicle Simulation Branch
Naval Air Warfare Center Aircraft Division
Patuxent River, Maryland 20670

Abstract

Simulation has long been an accepted training tool, but its acceptance as a flight test enhancement or alternative has been much more difficult to attain. The process of developing a validated, high-fidelity simulation is an important step in this acceptance because it builds confidence in the quantitative data that flight test support demands. This confidence opens up a new world of opportunity to support a variety of projects and explore new frontiers. The F/A-18 simulation at the Manned Flight Simulator (MFS) has been used for many of these projects. Two of the most recent and unique are discussed in this paper. One of these involved connecting the F/A-18 simulation via an encrypted data link to the Real Time Processing System (RTPS) facility, which is the telemetry ground station for flight testing at Patuxent River. The simulation was modified to use this link to provide flight test engineers with critical safety of flight data in real time to prevent departure from controlled flight. This data was computed by the simulation using the telemetered aircraft data. On another project, the simulation was used off line to determine if a correlation between uncommanded roll rates at various air speeds and the presence of a specific misrigged surface could be found. The results have been used by the Navy to return many "bent" aircraft with significant reductions in mission capability to nominal flight status.

Introduction

In the current fiscal environment, the search for low cost alternatives for the flight time needed to test a new platform or system is a driving concern. The use of high-fidelity simulation can significantly reduce the number of flights, and the associated cost, required to complete a program. Simulation can also be used to perform and evaluate flight test maneuvers that are much too dangerous to be tested in the real aircraft. However, before a simulation can be used to enhance flight testing, it must gain the confidence of the pilots and flight test engineers. Program managers must also be thoroughly convinced that it has the required fidelity since decisions they make based on the results can affect costs and safety.

An engineering simulation that is used to support flight test is an entirely different entity than a training simulation. This must be taken into account when assessing the required fidelity of each modeled subsystem. A trainer must feel like the real aircraft qualitatively, and all of the pilot interfaces must work exactly as on the aircraft. On the other hand, an engineering simulation must be quantitatively accurate with all of the primary displays and interfaces functional, but secondary displays and functions are of lesser importance. Therefore, available funding is spent on engineering simulations to accurately model functions such as aerodynamics, propulsion and flight controls, while functions such as environment, communications, engine start, etc. are of far less importance.

The F/A-18 simulation at the MFS is successfully used on a regular basis to support flight test programs at Patuxent River. A better understanding of the usefulness of high-fidelity simulation in flight testing can be gained through an examination of the process used to develop the required fidelity, the challenge of achieving user acceptance, and a discussion of specific examples of successfully supported programs by the MFS.

The MFS Facility

The F/A-18 simulation is a part of the MFS facility[1]. It can be used in any of five lab stations: a 40 ft dome, a 6 degree of freedom motion base, a helmet mounted display station, or two engineering development stations as shown in Figure 1.

The station used is determined by the project: cruise work is normally done in the dome, while take-off and landing work is normally done in the motion base. The helmet mounted display station is being used on a

* Aerospace Engineer, AIAA Member

more regular basis for a variety of tasks. The cockpit requires only 15 minutes to swapout due to the modular design of the facility and the standard interface on the back panel of the cockpit.

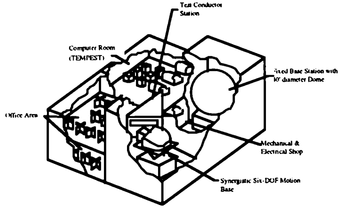

Test Conductor Station
Computer Room (TEMPEST)
Fixed Base Station with 10' diameter Dome
Office Area
Mechanical & Electrical Shop
Synergistic Six-DOF Motion Base

Figure 1 - MFS Facility

The MFS facility is part of the Air Combat Environment Test and Evaluation Facility (ACETEF). This grouping of interconnected labs is a resource greater than any of the labs on their own. The facility is a fully integrated test facility developed to support multispectral test and evaluation of aircraft and aircraft systems in a secure and controlled environment.

The F/A-18 Simulation

The simulation can be run off-line in desktop mode or in a realtime pilot in-the-loop configuration in one of the lab stations. The F/A-18 simulation being run in a representative configuration is shown in Figure 2.

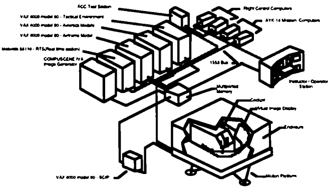

FCC Test Station
VAX 4000 model 60 - Tactical Environment
VAX 4000 model 60 - Avionics Models
VAX 4000 model 60 - Airframe Model
Motorola 88110 - RTS (Real time station)
COMPUSCENE IVA Image Generator
Flight Control Computers
AYK-14 Mission Computers
1553 Bus
Instructor / Operator Station
Multiported Memory
Cockpit
Virtual Image Display
Enclosure
VAX 4000 model 60 - SCIP
Motion Platform

Figure 2 - Representative F/A-18 Lab Station Setup

The simulation is linked with the lab resources using the Simulation Control Executive (SCE) software developed in-house at MFS. Depending on the project, the SCE will configure the simulation to include a variety of options. These options include running with Flight Control Computers (FCC's) and Mission Computers (MC's) in-the-loop via the 1553 bus, and connecting to any of the local ACETEF labs. While these and many other options are available, the strength of the simulation lies in the accuracy of the airframe models.

The F/A-18 simulation has been upgraded to a point where it provides very high fidelity and results that have a high degree of confidence. The upgrades have been concentrated in the areas of aerodynamics, flight controls, actuators, weight & balance, and propulsion.

The most extensive upgrades have been to the aerodynamic math model[2]. The new aerodynamic data tables incorporate the results of parameter identification (PID) analysis as well as the aerodynamic increments associated with rotation about the velocity vector (rotary balance wind tunnel data). The result is an aerodynamic data base continuous in angle-of-attack from -90° to 90°, sideslip from -30° to 30°, and Mach number from 0 to 2, with representative modeling of the low-speed envelope for both the single-seat and two-seat aircraft.

To validate the simulation a dedicated flight test program of 30 flights was used to gather validation data, and data that could be used for future upgrades[3]. This validation data is very important as it quantifies the accuracy of the simulation. In addition, piloted qualitative evaluations are important to get an idea of the fidelity of the simulation, the associated transport delays, and to get pilots comfortable with its use as another test tool. The cost of these thirty flights is quickly recovered by reducing the number of flights required for future flight tests because of the increased use of the simulation.

Currently, the simulation is used heavily for flight test support, accident investigations, flight controls and avionics work. Two recent projects include support for an asymmetric stores envelope expansion flight test program, and support of a program to identify misrigged surfaces on the F/A-18.

Asymmetric Store Loading Envelope Expansion

In a unique application of simulation, the MFS F/A-18 simulation was used to provide real-time feedback of critical aerodynamic and flight control data to flight test engineers during a test flight for the purpose of risk reduction. To accomplish this, a recently developed link between the simulator and telemetry station was used. The MFS was linked to the Real

Time Processing System (RTPS) via a T1 line with encryption/decryption capabilities[4]. This link shows great potential to improve flight testing by using the simulation in a variety of ways - only one of which will be examined here.

MFS - RTPS Link

A high level pictorial representation of the configuration used for the asymmetric stores envelope expansion program is shown in Figure 3.

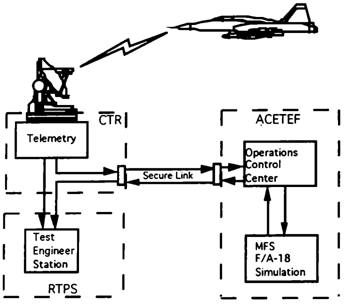


Figure 3 - MFS - RTPS Link

Nominal safety of flight and test data are telemetered, saved and displayed at the RTPS ground station. In addition, certain data is transported via the secure link to the MFS. For this application, the F/A-18 simulation was modified to allow the incoming variables to overwrite the simulation variables and control the simulation. Aircraft states, atmospheric conditions, pilot inputs, etc. come from the actual aircraft and the simulation tracks the real aircraft. In return, the simulation sends information back to RTPS that would be otherwise unavailable. This information includes data that can not be accessed or determined on the aircraft, and data that is specially calculated using the simulation.

Simulation Generated Data

For this program, the challenge was to find a way to graphically represent the amount of roll control power the aircraft had at any point in time. Specifically - when is the pilot in danger of not having enough roll authority to counter the asymmetry. Two methods were developed to address this problem: (1) determine the ratio of current rolling surface deflection against the maximum possible at the current flight condition, and (2) determine the ratio of the current rolling moment due to the asymmetry against the moment that would be generated by an opposing full roll input. The numerators are determined directly from aircraft data, while the denominators are simulation derived quantities.

This task is more complex then it may appear at first. There are a total of 8 surfaces that work in concert to produce rolling moments. The digital flight controls determine the mix of these surface deflections and deflection limits. The limit on each individual surface is a complex series of functions that are dependent on the flight conditions.

Surface Deflection Method

This method was developed to give an indication of the roll control surface travel that is still available for a given flight condition. Because of the number of surfaces involved and the non-linearity of the aerodynamics, this method is not useful for determining actual roll power available. It is, however, useful in knowing what percentage of the rolling surface's deflection remains. The simulation returns the percentage of rolling surface used as follows:

$$\text{Roll\_surface}_{\text{used}} = \frac{\Sigma\text{rolling\_surface\_deflections}_{\text{actual}}}{\Sigma\text{rolling\_surface\_deflections}_{\text{max}}}$$

where Σroll_surface_deflections$_{\text{actual}}$ is the summation of differential stabilator, aileron, leading edge flap (LEF), and trailing edge flap (TEF) from the aircraft. Σroll_surface_deflections$_{\text{max}}$ is the summation of the maximum allowable deflections of these same surfaces at the given flight condition. These maximum deflections are calculated using the simulation. To calculate these values the roll axis control law subroutines were executed using all telemetered data except that either a full left stick or full right stick input was used depending on the attempted direction of roll. The dynamic filters were removed from the control laws so that the control surface deflection limits could be determined immediately. A simple block diagram of the simulation modifications is shown in Figure 4.

Rudder pedal inputs were unnecessary because the F/A-18 has a full authority rolling surface to rudder interconnect (RSRI).

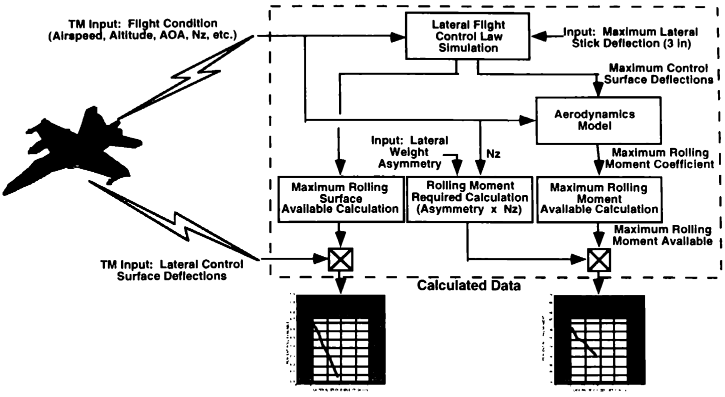

TM Input: Flight Condition (Airspeed, Altitude, AOA, Nz, etc.)
Lateral Flight Control Law Simulation
Input: Maximum Lateral Stick Deflection (3 in)
Maximum Control Surface Deflections
Aerodynamics Model
Input: Lateral Weight Asymmetry
Nz
Maximum Rolling Moment Coefficient
Maximum Rolling Surface Available Calculation
Rolling Moment Required Calculation (Asymmetry x Nz)
Maximum Rolling Moment Available Calculation
Maximum Rolling Moment Available
TM Input: Lateral Control Surface Deflections
Calculated Data

Figure 4 - Simulation Calculated Parameters

The delay between the calculation of the data and the display to the test engineer was 0.1 seconds due to the synchronization of the simulator and telemetered data.

Rolling Moment Method

The ratio of rolling surface deflection, while useful, does not give a true picture of the roll power. To get a clearer picture, the rolling moment method takes the surface deflection method one step further. All of the surface deflections that were calculated for the full stick input were sent to a modified version of the aerodynamics routine. This routine calculates the roll moment coefficient and thereafter the rolling moment generated by the full stick input. This is the maximum rolling moment the pilot can generate at that particular flight condition.

Rather than comparing this moment to the current total rolling moment, it is compared to the current rolling moment due to the asymmetry. A better representation of loss of roll control is presented in this manner. The current asymmetry is calculated at RTPS (either 0 or the preflight determined value depending on jettison) and sent to the simulation as part of the data stream. This asymmetry is then multiplied by the filtered normal acceleration signal from the aircraft FCC accelerometer. The result is the rolling moment due to the asymmetry.

The maximum rolling moment that can possibly be generated is calculated by the simulation in a manner similar to the surface deflection method as shown in Figure 4. The difference is that the surface deflections that were determined are put through the aerodynamics model to come up with the rolling moment coefficient. This is converted to a rolling moment that is the maximum possible at the given flight conditions.

These two values determine the ratio of rolling moment used according to the equation:

$$\text{Roll\_Moment}_{\text{used}} = \frac{\text{Roll\_Moment}_{\text{asymm}} \times \text{Normal\_Load}}{\text{Roll\_Moment}_{\text{max}}}$$

This value is sent to RTPS for display to the flight test engineer. It is the most important simulation generated value sent.

RTPS Use of the Simulation Data

After being sent to RTPS the data is used as part of a display used for real-time departure prevention. Presented in Figure 5 are the parts of the display that use the simulation data. This display is the first of its kind at NAWCAD to merge simulation and aircraft data to be used real-time for flight test risk reduction.

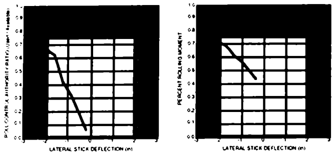


Figure 5 - Simulation Generated Displays

Both simulation calculated variables are presented in similar fashion as shown above. They are plotted against lateral stick so that a quick comparison of pilot command and effect is apparent. The outer limits of each box represent full left and right stick on the horizontal axis, and either roll moment ratio or surface deflection ratio from 0 to 1 on the vertical axis. The light inner box represents the safety factor that was applied for safety of flight calls. These factors were determined to be 75% for the ratios, and 2 inches (or 2/3) for lateral stick. The display used the current point plus the past few points to show the trend in a glance.

This use of simulation to enhance flight test data was very successful. The displays presented important information in a manner that was quickly digestible, and actually allowed engineers to get comfortable interpreting trends. This information made the display useful not just for safety of flight, but also for a conceptual understanding of what was going on throughout the maneuvers.

Correcting Aircraft Control Surface Rigging

Many F/A-18 fleet aircraft currently in use suffer from uncommanded rolling moments. These moments are generated by control surfaces that have either become misshapen or have warped due to age or have been improperly rigged. On some aircraft these moments are so large as to create a departure hazard at higher angles of attack. These moments can normally be eliminated if it can be determined which surface (or surfaces) is creating the moment. Unfortunately, the highly augmented nature of the flight control laws makes it very difficult to isolate the surface that is creating the moment. In the past, some of the aircraft were improved by the contractor using a trial and error method approved for rerigging. The requirement now was to develop a formal yet simple procedure to identify the misrigged surface and the action required such that it could be routinely performed in the fleet by persons with no specialized training.

The F/A-18 simulation was used to help address this problem. By working backward from simulated known misrigged surfaces, a search was made for a pattern that would help identify the offending surface. On each control surface of the aircraft, various misrigs were added and the simulation trimmed at various airspeeds. The simulation was then run to get the local roll rate maximum. This analysis was done in the cruise, full flaps, and half flaps configurations. Only the cruise analysis will be presented here as the other configurations were analyzed in the same manner. The test matrix for the cruise configuration is shown in Figure 6. For this program, a matrix as sparse as 80 points was all that was necessary to identify the trends in cruise configuration. The matrices for PA and PA1/2 were the same except speeds of 150, 175, 200, and 250 knots were used. In practice the cruise configuration is used on the actual aircraft.

Surfaces misrigged	Aircraft speeds	Amounts of misrig
Ailerons	200 kts	0.5 deg
Stabilators	300 kts	1.0 deg
Rudders	400 kts	1.5 deg
LEF's	500 kts	2.0 deg
TEF's		

Figure 6 - Cruise Misrig Variables

To allow implementation of the test matrix, the production simulation was modified so that a misrig could be added to any control surface by adding a constant bias to the control surface position at the actuator model output. This biased position was then used for all downstream calculations, including aerodynamics. This implementation gave the most physically representative system.

A representative run is shown in Figure 7. The simulation was trimmed at 400 knots and 20,000 feet. The stabilator was then misrigged 2 degrees and the simulation was run. Notice that a steady roll rate was achieved within half a second. This roll rate was recorded and the process repeated for all points.

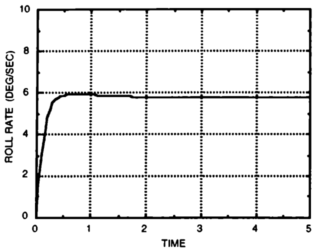


Figure 7 - Stabilator Misrig Run

The series of graphs in Figure 8 shows the results of the simulation analysis of misrigs for the cruise configuration. Note that the characteristic curve for each surface is shown. The actual graphs are a series of curves that are similarly shaped but differ in magnitude with the magnitude of the misrig.

This data shows that there is a unique relationship between uncommanded roll rate and airspeed for a misrig of each surface. Therefore, these graphs can be the basis for designing a flight test profile to identify flight control surfaces on the actual aircraft that are causing the uncommanded roll rates.

A simple flight test method has been developed to compare aircraft to the simulator data and identify a misrigged surface[5]. This method requires that the pilot take off and transition to cruise configuration without using lateral trim, stabilize at 200 knots, release the stick, and measure the time it takes to roll 30 degrees. He then repeats this at 300, 400, 500 and 550 knots. Once back on the ground, the time to roll can be converted to an average roll rate and plotted against airspeed as with the simulator data. A simple match between the aircraft trace and one of the surface traces is usually obvious. The fact that the simulator used maximum roll rates and the flight test uses average roll rates is not an issue as the curves are the same shape with minor magnitude differences.

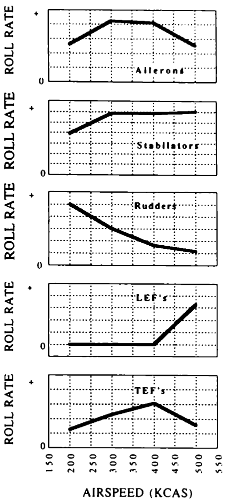


Figure 8 - Misrigged Surface Representative Curves

The problem is slightly more complicated if more than one surface is misrigged. Usually one of the surfaces is the dominant contributor, and one of the trends from the simulator can be identified. This surface is dealt with first, then the flight test is repeated. Once the first surface is corrected, the second surface becomes clear. In the simulator, pilots have been able to identify up to three simultaneous misrigs. In practice there has not been the need to identify more than two simultaneous surfaces.

This technique has been highly effective in correcting control surfaces on many F/A-18 fleet aircraft that were exhibiting uncommanded roll rates[5]. The extension of this technique to other highly augmented aircraft, is a simple, cost effective way to improve the flying qualities of our current assets.

Conclusions

These examples of the use of simulation to support flight test are only two of many that the MFS uses on a daily basis to support its customers. Simulation is a valuable tool, and as such is limited only by the imagination and creativity of the engineers that use it. There is no program that cannot be supported in some way by utilizing simulation.

The key to utilizing any simulation in such manners as the MFS is the faith inherent in the fidelity of the simulation. A thoroughly validated simulation is the cornerstone of any effort.

In the past, simulation was used almost exclusively in a stand alone manner. While this is still useful today, using simulation in concert with other tools can multiply the effectiveness of the effort.

The following are lessons that have been learned:

• Refine the simulation to get the highest possible fidelity given the financial reality. Any money spent early refining the models will be returned many times in the future with savings from the programs which will depend on the simulation. With the abilities of today's fighter aircraft, aerodynamic databases must be expanded beyond the current standard - beta ranges need to be extended to ±90° and alpha beyond this. This is to support high angle of attack, thrust vectoring, and out of control flight regimes.

• Verify the simulation both qualitatively and quantitatively and have the results readily available for the customer.

• Convince users of the fidelity of the simulation, AND make sure they know the limitations and constraints which they must obey to obtain valid results.

• Simulation is one of many tools - find other available tools that compliment the simulation and use them in concert.

• Work closely with customers and seek their input - they often have ideas that aren't constrained by simulation paradigms.

• If you spend the money and effort to develop a first rate simulation, users will seek you out with ideas on how they can use it.

References

1. Burton, R., Miller, C., Mills, R. "Manned Flight Simulator and the Impact on Navy Weapons Systems Acquisition"; AIAA Flight Simulation Technologies Conference Proceedings, Aug. 1994, pp. 163-170

2. Fitzgerald, T., Ralston, J., Hildreth, B. "Improvements to the Naval Air Warfare Center Aircraft Division's F/A-18 Subsonic Aerodynamic Model"; AIAA Flight Simulation Technologies Conference Proceedings, Aug. 1994, pp. 1-9

3. Bonner, M., Gingras, D. "Status of a Comprehensive Validation of the Navy's F/A-18 A/B/C/D Aerodynamics Model"; AIAA Flight Simulation Technologies Conference Proceedings, July 1996

4. McNamara, W., Stanley, P., Nichols, J. "RTPS Telemetry - Simulator Link at the Naval Air Warfare Center"; International Telemetry Conference Proceedings, Las Vegas, 1995

5. Traven, R., Whitley S., Frazier F.A. "Developing Flight Test Techniques to Ensure Proper Rigging of Highly Augmented Aircraft"; Society of Experimental Test Pilots 39th Symposium, Sep 1995, pp. 171-179

AIAA-96-3521-CP

IMPLEMENTING FANS IN THE NASA 747-400 FLIGHT SIMULATOR FOR AIRSPACE OPERATIONAL RESEARCH

Barry T. Sullivan, NASA Ames Research Center *
Ramesh C. Panda, NSI Technology Services Corp. **
Paul A. Soukup, NSI Technology Services Corp. ***

ABSTRACT

This paper describes the implementation effort for integrating the FANS-1 upgrade into the NASA Ames Research Center's 747-400 flight simulator. FANS is an advanced avionics system upgrade that will utilize global based satellite information and advanced technology to provide communications, navigation, surveillance and air traffic management for the twenty-first century. This paper describes the various upgrades and enhancements that were made to the NASA 747-400 simulator and how these changes will enable NASA to support important national research programs that will further utilize these advanced features and capabilities provided by FANS. These changes include modifications to the 747's advanced avionics systems, flight displays, warning system, and experimenter control stations. New features as a result of integrating FANS include a Global Positioning System (GPS) simulation, an enhanced satellite based communications system, new ground based air traffic controller and airline operational communications features, as well as new tracking capabilities and functions. In addition, this paper describes some of the upcoming research programs and benefits that may be gained as a result of FANS.

INTRODUCTION

The current system for managing air traffic in the oceanic arena was developed and implemented many years ago, when air travel was substantially less congested than it is today. This system used procedural controls with air traffic control intervention via High Frequency (HF) radio communications. During that time, this system was very safe, reliable and efficient. Since then, technology has evolved and now high speed jet propelled aircraft fly along very congested and inefficient airways. In addition, current HF radio communications have become heavily congested resulting in a system that is more susceptible to potential operator errors due to problems interpreting information that is being transmitted via the radios.

Through the efforts of the International Civil Aviation Organization (ICAO), the International Air Transport Association (IATA), and other air traffic service providers, a Future Air Navigation System (FANS) concept was developed to transition the current air navigation system to a more advanced, efficient and safer system utilizing global satellite based information for performing Communications, Navigation, Surveillance and Air Traffic Management (CNS/ATM) functions for all regions of the world for the twenty-first century.

The FANS concept will make extensive use of automation, satellites and data communications to provide the future air traffic services for the new CNS/ATM system. The aeronautical communications system in the new CNS/ATM will make use of digital communications to provide an efficient means of passing information. While the need for voice communications will remain, the introduction of data communications will enable fast exchange of information between all parties via a single network. The increasing use of data communications between aircraft and various ground systems will require a communications system that gives users close control over the routing of data, and allows different systems to communicate with each other without human intervention. This system will have to support two-way pilot-controller data communications in addition to the present day voice communications. The global inter-networking infrastructure that will be used to support the FANS concept is the Aeronautical Telecommunications Network (ATN). This virtual network is comprised of existing and planned telecommunications networks and will link the various air-ground and ground-ground data systems together. The ATN will operate globally, encompassing all aeronautical data communications associated with the international aviation environment.

The ATN is composed of three basic sub networks which include a ground network, an air-ground network and an airborne network. The ground network consists primarily of the Aeronautical Fixed Telecommunications Network/Common ICAO Data Interchange Network (AFTN/CIDIN) and airline industry private networks such as the ARINC Data Network Service (ADNS) and the SITA network. The air-ground network contains the satellite, Very High

* NASA 747-400 Simulator Manager, AIAA Member
** 747-400 FANS Lead Project Engineer
*** 747-400 Simulator Project Engineer

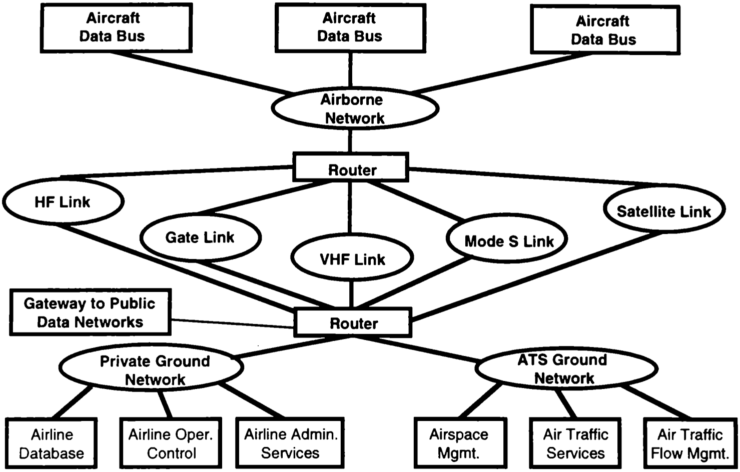


Figure 1 - Aeronautical Telecommunications Network (ATN)

Frequency (VHF), Mode S, HF and gatelink sub networks, while the airborne network consists of the airborne network router and the aircraft data busses. Figure 1 shows the various networks and how they are linked within the ATN.

The Global Navigation Satellite System (GNSS), a key feature of FANS, will provide navigation coverage and accurate time referencing on a global basis. This system includes one or more satellite constellations, aircraft receivers, ground monitor stations and system integrity monitoring. GNSS provides users with the capability to perform on-board position determination referenced to a standard geodetic reference system, independently from its geographic location. The GNSS will make use of the United States' Global Positioning System (GPS) and the Russian Federation's Global Orbiting Navigation Satellite System (GLONASS). Each of these systems comprises a constellation of satellites including spares. In the future, ground based augmentation is expected to extend GPS capabilities by providing the capability to conduct precision approaches. With the introduction of the GNSS, airplanes will be able to fly routes efficiently and accurately.

As part of the FANS concept, surveillance will be the basic method by which controllers will monitor aircraft separation, manage airspace efficiently, and assist pilots in navigating their aircraft safely. Three tools critical to surveillance under FANS include an Airborne Collision Avoidance System (ACAS), Automatic Dependent Surveillance (ADS), and Secondary Surveillance Radar Mode Select (SSR Mode S). ACAS will provide back-up to the air traffic services by alerting flight crews of potential collisions through an on-board alerting system that is independent of any ground based system (presently implemented in the U.S. as Traffic Alert and Collision Avoidance System - TCAS). ADS will support automatic surveillance of appropriately equipped aircraft independent of any pilot-controller communications. For oceanic surveillance/tracking, ADS will match land capability through the use of inertial navigation or GNSS to determine aircraft position. Satellite communications will automatically transmit aircraft position and other information in near real-time. Over land regions ATC Radar Beacon System (ATCRBS) and Secondary Surveillance Radar (SSR) will continue to be used for tracking/surveillance.

The main beneficiary of the new CNS system will be the future Air Traffic Management (ATM) System. New CNS systems will provide more and better information transfer between ground and aircraft systems during all phases of flight and between all Flight Information Regions (FIRS). Although the future ATM system will be more efficient and enhanced, its basic functions will remain the same. These functions include Air Traffic Flow Management (ATFM), Air Space Management (ASM) and Air Traffic Services (ATS) which encompasses flight information services, alerting services, and air traffic control. These systems will be optimized strategically and tactically to expedite and maintain a safe and orderly flow of traffic while giving due consideration to costs of implementation and system operation. In addition, they will enable aircraft operators to meet their planned times of departure and arrival and adhere to their preferred flight profiles with minimum constraints and without compromising agreed levels of safety.

Many of the FANS concepts are currently being implemented and tested in the aviation community. It is expected that it will take about fifteen to twenty years before it is fully implemented. As a result of this gradual implementation there will be many human factors and airspace operations issues that will need to be addressed.

At the NASA Ames Research Center, a unique national research facility is available to support the kinds of research issues that will result by implementing FANS. The NASA 747-400 Flight Simulator, located at the Crew-Vehicle Systems Research Facility (CVSRF), at Moffett Field, California is prepared to help address these issues. The NASA 747-400 Simulator is but one integral component of the CVSRF. This unique facility was built in the early 1980's, primarily to support aviation human factors research. Over the years, the CVSRF's role has also expanded to include studying issues pertaining to airspace operations, in an effort to try and find ways of improving aviation safety and increasing the efficiency of the national airspace system. Through the use of it's full-mission simulation capability the CVSRF has positioned itself to support such issues. In addition, the CVSRF includes an Air Traffic Control (ATC) simulator, which can be integrated with the 747 simulator, providing the air-ground communications and creating a "full-mission" environment with the ability to perform complex multi-aircraft full-mission scenarios to support human factors and airspace operations research.

This paper focuses on the development and integration effort involved with implementing FANS in the NASA 747-400 Simulator. It highlights the various new features introduced by FANS, as well as the numerous modifications made to the 747 simulator and it's related systems in support of this upgrade. In addition, it describes some of the upcoming research programs and benefits that may be gained as a result of implementing FANS.

OVERVIEW OF 747-400 SIMULATOR

The NASA 747-400 flight simulator is an exact replica of a United Airlines 747-400 airplane cockpit. It is modeled after aircraft number RT612 and was built by CAE Electronics Ltd. in St. Laurent, Canada. The simulator represents a state of the art current technology glass cockpit. The NASA 747-400 simulator is certified to both FAA Level D and the newly established International Qualification Standard Level II, as established by ICAO. It is equipped with a Flight Safety International VITAL VIIe visual system, a CAE series 600 digital control loading and motion system, and an IBM RISC 6000, Model 580 host computer. The simulator is also equipped with unique research capabilities which make it an ideal tool for conducting aviation human factors and airspace operations research. These capabilities include reprogrammable flight displays, cameras mounted throughout the cockpit to allow videotaping of pilots actions or images of the various flight displays or Multifunction Control Display Units (MCDUs) within the cockpit, and a separate IBM RISC 6000 Model 560 computer for data collection and reduction purposes. In addition, user friendly touch sensitive experimenter operator control stations are also included, as well as repeater monitors of each of the cockpit displays. The simulator is also integrated to the CVSRF's ATC simulator and is networked to other research facilities such as the FAA Technical Center's ATC simulation complex in Atlantic City, and the FAA's Aeronautical Test Facility in Oklahoma City. These specially adapted capabilities provides NASA with a unique research facility for examining operational and human factors issues pertaining to the implementation and utilization of FANS. The relative importance of integrating FANS in the NASA 747-400 simulator is that it enables the CVSRF to support important national research programs.

FANS FUNCTIONS

The FANS-1 upgrade is a partial realization of a futuristic CNS/ATM concept envisaged by ICAO. This implementation is better described in terms of FANS functions which are discussed here as a prelude to the airborne and ground based upgrades. Components belonging to a majority of these functions exist both in the airborne and ground based systems.

AIR-GROUND DATA-LINK COMMUNICATIONS

Digital data-link for air-ground communications is already in widespread use today. A discussion of the systems and the support infrastructure that currently exists is useful in understanding the ICAO proposed schemes for FANS data link.

The ARINC Communications Addressing and Reporting System (ACARS), which dates back to the late seventies was initially adopted by the major airlines as a logging system for flight phase times for their flights. Known as the OOOI (Out-Off-On-In), these reports transmitted the aircraft push-back, takeoff, landing and at-gate times to a central logging system at the airline operations center. Today, the airlines use the system for a number of other types of information gathering and reporting jobs.

The end-to-end systems in this setup consist of the ACARS Management Unit (AMU) on the airborne side connected via a network of VHF stations to the ground operations center. The VHF enroute coverage is available across the continental U.S. and several areas around the world. In addition, local coverage is available at most airports. For aircraft equipped with SATCOM, the coverage is available globally since at present, the network service providers also support ACARS communications via satellite. In a typical glass cockpit installation like the 747-400, the AMU may be connected to several aircraft systems which automatically collect data and send it to the appropriate ground station without imposing any additional workload on the flight crew. The Aircraft Condition Monitoring System (ACMS), for example, can collect engine data and send to the aircraft maintenance operations. The variety of information the current ACARS systems can provide includes information on weather, winds aloft, flight planning and position. Many airlines also use ACARS for Pre-Departure Clearances which helps reduce congestion over the voice channels.

The ACARS data-link communications are based on a fixed length character based message block, as per the ARINC 618 protocol. Each character byte is made up of a 7-bit ISO-5 character with the 8th bit providing odd parity. The 256 character message block is made up of a header and text components. The information in the header includes the aircraft identification while the 220 character text portion is used to transmit the message data. The end of the message is appended with a 16 bit checksum. Multiple blocks may be utilized to transmit messages that do not fit into a single 220 character text portion. A special end of the message block bit is used to inform whether additional blocks follow.

FANS DATA-LINK APPLICATIONS

Under the ICAO FANS concept, all data-link communications are expected to be transitioned to a digital network based on the International Standards organization (ISO) Open Systems Interconnect (OSI) standards. Known as the Aeronautical Telecommunications Network (ATN), this wi̇̃ll encompass all air-ground and ground-ground aviation data communications. The VHF, SATCOM, Mode S and possibly a future HF data network will be sub-networks under the ATN. The ATN compliant applications will, however, need to be based on bit-oriented, data transparent message based protocols. Unfortunately, the current ACARS data is character oriented and incompatible with ATN. Since the ACARS system is expected to be in use for some time, the aviation industry has come up with an interim solution which allows bit-oriented applications to communicate over the ACARS network. This standard, called ARINC 622, defines the ACARS Convergence Function (ACF) which performs the bit-to-character and character-to-bit conversions required in order for bit-oriented applications to be able to use the character oriented ACARS network. The ADS and the ATC DL under FANS have been defined as bit oriented applications and already take advantage of this interim standard.

AIR TRAFFIC SERVICES FACILITIES NOTIFICATION (AFN)

An Air Traffic Services (ATS) facility becomes aware of a FANS data-link equipped aircraft by means of the AFN data-link function. As is the case with the other data-link applications, there are airborne and ground ends to the AFN, with the airborne side residing in the flight management computer (FMC).

The AFN Initial Notification data transaction is normally initiated from the aircraft with the aircrew member entering a ground ATS facility identification through a logon page on the MCDU. A downlink contact message is then sent to the ground facility where it is acknowledged by sending an appropriate uplink response. These two messages exchange all the addressing information that will be required for the air and ground application counterparts of the ADS and ATC DL applications to establish their respective communication links.

When the aircraft moves from one region to the next, the addressing information is exchanged by the controlling facilities through ground-ground links. As an alternative, a ground facility may utilize the aircraft AFN to pass on the information to the next facility. An additional 3 data-link messages, constituting the Contact Advisory transaction are provided for this purpose.

AIR TRAFFIC CONTROL DATA-LINK

Once an AFN Initial Notification transaction is successfully completed, a ground ATS facility may establish a data-link connection between the ground based and airborne elements of the ATC DL application. The ATC DL, also known as the Controller Pilot Data-Link Communications (CPDLC), consists of a set of uplink and downlink messages used for controller-pilot dialogue. The message set, specified by RTCA DO-219 is based on phraseology currently in use for voice communications. A typical ATC DL message consists of a fixed message portion and one or several variable fields. For example, in the vertical clearance CLIMB TO AND MAINTAIN FL330, altitude is the input variable. The various types of input variables include time, speed, altitude, position, vertical rate, direction, distance, facility identifications, frequencies and route clearances. The messages are grouped into the functional categories as follows:

Uplink Message Groups:

- Responses / Acknowledgments
- Vertical Clearances
- Crossing Constraints
- Lateral Offsets
- Route Modifications
- Speed Changes
- Contact / Monitor / Surveillance Requests
- Report / Confirmation Requests
- Negotiation Requests
- Air Traffic Advisories
- System Management Messages
- Additional Messages

Downlink Message Groups:

- Responses
- Vertical Requests
- Lateral Offset Requests
- Speed Requests
- Route Modification Requests
- Negotiation Requests
- Emergency Messages
- System Management Messages
- Additional Messages

ATC DL application sends and receives only bit oriented messages. The ACF, defined by ARINC 622, is currently included at each end to accomplish the conversion to and restoration from the character based ACARS format in which the actual message is transmitted.

AIRLINE OPERATIONS COMMUNICATIONS DATA-LINK

A separate data-link function dedicated to airline communications is provided by FANS. The AOC DL, as it is called, in a sense is an extension to the functions currently supported via the AMU. The airborne side of the AOC DL application, however, resides in the FMC. Some of the new features in AOC DL include the ability to uplink a flight plan which can be automatically loaded by the flight crew. Similarly, enroute and descent forecast winds can be separately uplinked. A flight crew can also send a request for the above data via a downlink to ground operations. Other features supported in AOC DL include automatic triggering of position reports and winds aloft without any flight crew input. The AOC DL application currently utilizes character based messages.

AUTOMATIC DEPENDENT SURVEILLANCE

Automatic Dependent Surveillance (ADS) provides aircraft current position, intent and other information derived from various on-board avionics systems via an air-ground data-link, typically to a requesting ATS facility. A contract initiates and establishes the type of information to be transmitted. The airborne ADS function, resident in the FMC, assembles and delivers the required information in response to the contract received from its ground-based counterpart. The contract uplink and the associated downlink report transaction takes place without any flight crew intervention. The ADS is expected to fulfill a very important role in the future as an automatic surveillance mechanism, especially in the oceanic environment where there is no radar coverage.

Information supplied by the ADS can be periodic. on-demand, or event driven. In a periodic contract, the requested data is updated at the interval specified in the contract uplink. For on-demand contracts, the requested report is sent once, while in the event contract the reports are generated when any of a set of pre-defined events takes place. The ADS reports are divided into groups of data and include:

- Basic - latitude, longitude, altitude, time, position fixing accuracy, navigation redundancy.
- Flight Identification
- Earth Reference - true track, ground speed, vertical rate.
- Air Reference - true heading, mach speed, vertical rate
- Meteorological - wind speed, direction, temperature
- Predicted Route - latitude, longitude, altitude at the next and next+1 waypoints in the active flight plan and estimated time of arrival at the next waypoint.
- Intermediate Projected Intent - up to 10 waypoints in the active flight plan, specified as along-track distance, track, altitude and time where a change to the target altitude, speed or course is expected.
- Fixed Intent - projected waypoint in the active waypoint specified as latitude, longitude, altitude, given a projected time.

A contract may request a report for a single group or a combination of groups, either on a periodic or on-demand basis. For an Event contract, the airborne ADS reports on any of the following occurrences:

- Waypoint Sequence - Basic and Predicted Route group data is supplied anytime a waypoint is sequenced.
- Altitude Range - Basic group data is supplied when the aircraft violates the specified altitude range.
- Vertical Rate - Basic and Earth Reference group data is supplied when the specified vertical rate threshold is exceeded.
- Lateral Deviation - Basic group data is supplied when the lateral deviation exceeds the specified threshold.

In addition to the ADS contracts discussed above which are generally initiated by a ATS facility, the flight crew may also initiate an emergency periodic report which transmits the Basic group and other data at pre-determined intervals. A complete definition of ADS is available in ARINC 745-2 and RTCA DO-212.

REQUIRED NAVIGATION PERFORMANCE

The advent of highly accurate satellite navigation systems has resulted in the development of new concepts to better utilize the available airspace with increasing levels of traffic. Required Navigation Performance (RNP) is one such concept.

Although RNP affects both the aircraft and the airspace in which it is operating, it is essentially a parameter specifying the navigation performance requirements that the aircraft must satisfy in order to operate in that airspace. The airspace in this context can be a segment of an airspace, a navigation flight phase or a procedure. A rigorous definition of RNP is beyond the scope of this paper and the interested reader is referred to reference [8]. For the current discussion, it is sufficient to say the RNP type may be thought of as a containment surface within which the aircraft is expected to stay with a probability of 95%. As an example, an aircraft operating in a RNP 12.0 airspace is expected to stay within a 12 nautical mile radius 95% of the time. By extending this containment surface along an approach segment, an operational envelope of airspace or "tunnel in the sky" is defined for flying the approach. In the geometry of this tunnel, individual requirements such as obstacle clearance or terrain separation can also be factored in. The RNP function contained in the FMC includes a set of default values for the various operational flight phases. The default values can be overridden by manual input on the Multi-function Control Display Unit (MCDU).

REQUIRED TIME OF ARRIVAL

The Required Time of Arrival (RTA) function in the FMC allows for placement of a time constraint at a single enroute waypoint. Time based, or 4-D guidance to the waypoint is then attempted by varying the enroute speed between minimum and maximum speed limits within the performance envelope. The FMC alerts the crew via a MCDU message when it cannot meet the time requirement.

The ability of an aircraft to control its arrival times at metering points enhances the effectiveness of ATC flow control. Utilizing the FANS data-link features, the controller may uplink required crossing times. This new capability allows aircraft to cross other routes, thus allowing access to airspace that would be otherwise restricted.

SIMULATED AIRBORNE ENVIRONMENT

On the Boeing 747-400 aircraft, the navigation system, the Flight Management System (FMS), the communications system, and the display system are affected by FANS. The navigation system is augmented by the addition of dual Global Positioning System (GPS) sensor units. The FMS has added capabilities such as using the GPS inputs to improve navigation performance, and to provide extensive two-way aircraft/air traffic controller and aircraft/airline operations communications. To support the additional communication features of the FMS, the communications system is improved by upgrading the ARINC Communications, Addressing, and Reporting System (ACARS) and adding a Satellite Communications System (SATCOM). The display system is upgraded to reflect the additions and improvements in the aforementioned systems.

GLOBAL POSITIONING SYSTEM

The Global Navigation Satellite System (GNSS) is currently composed of a constellation of GPS satellites that circle the earth continuously transmitting signals to sensor units on or close to the surface. The GPS satellites transmit accurate timing signals such as the Universal Time Clock (UTC) and other data pertaining to their exact location. The sensor units can accurately determine their position by comparing the timing of different GPS satellites. The accuracy of the position fix is dependent primarily on the Dilution of Precision (DOP) and the User Range Error (URE). The DOP accounts for the number of satellites in sight and their spatial distribution relative to the sensor. The URE reflects several factors that affect the satellite range measurement. The GPS sensor unit is equipped with a Receiver Autonomous Integrity Monitor (RAIM). The GPS RAIM monitors the integrity of the GPS satellites and eliminates any non-tolerant satellite source signal from the position calculation. The FANS-1 equipped 747-400 aircraft has two GPS sensor units which are directly connected to the two FMCs. The aircraft FMS uses the Inertial Reference System (IRS) as the primary source for navigation. Various external sources including radio signals such as Distance Measurement Equipment (DME) and localizers are used to update the IRS calculated position. With FANS, GPS becomes the main source for FMS position updating. Through the MCDU, the flight crew can interface with the GPS units. The latitude, longitude, and ground speed from both GPS units are displayed along with the active position update mode, UTC time, the current Actual Navigation Performance (ANP), and the RNP. The flight crew also has the ability to inhibit GPS updating altogether. The FMS RNP function provides display of ANP and RNP to the crew, and triggers alerts if the RNP is exceeded. The ANP is the radius that the current FMC position update mode can guarantee to within 95%. With GPS updating, a vastly improved ANP is obtained. The default RNP values range from twelve nautical miles in the oceanic environment down to 0.5 nautical mile in the approach phase. With ANP values below 0.5, FMS/GPS arrival procedures are therefore possible. FMS/GPS can be used as an additional method to guide the aircraft to any airport runway for nonprecision approaches. On oceanic routes, FMS/GPS navigation can reduce lateral and longitudinal separation as a result of the increased position and time accuracy.

The GPS simulation in NASA's B747-400 simulator is based on the Honeywell HG2021 Global Navigation Satellite Sensor Unit (GNSSU) for the B747-400 aircraft. The model simulates a dual-sensor system with each sensor sending data to both FMCs. GPS computed data are simulated by taking the simulator computed navigational data with random errors superimposed. If the GNSSUs have just been powered up, a system self-test and satellite acquisition delays are modeled. The state of the electrical busses and experimenter operator station (EOS) control variables are also used to determine whether the sensors are operational. Simulator interface modifications were required to model the circuit breaker for each unit. EOS control allows the user to effect the FMS ANP by imposing a sensor malfunction or a complete failure. If the GNSSUs are operational, appropriate outputs are computed in the host computer and sent to the FMCs once per second over a high speed ARINC 429 bus. ARINC 429 is based on a serial data communication protocol used to exchange data between different aircraft line replaceable units (LRUs). The outputs include latitude, longitude, ground speed, date and UTC, autonomous horizontal integrity limit, horizontal figure of merit, and GPS sensor status. These are the only sensor outputs that are utilized by the FMCs. The GPS simulation assumes a minimum required number of satellites are in view at all times.

AVIONICS SYSTEM UPGRADES

FANS modifications that apply to both the aircraft and the simulator include upgrades to the various installed avionics. Specifically, the 2 FMCs, the 3 MCDUs, the ACARS Management Unit (AMU), and the multi-input cockpit printer are affected by FANS.

The FMCs are the focal point for all the major airborne functions provided by FANS. The FMC upgrade is comprised of a memory expansion, new operational software and a new navigational database. FANS functions that are incorporated as a result of the upgrade are ADS, AFN, AOC DL, ATC DL, RNP, and RTA. Three program option select pins must be set to activate the FANS functions. For each FMC, they are the data-link enable, the ATS enable, and the GNSSU enable. Simulator interface modifications are also required to support the new GPS and program option select inputs.

The FANS upgrade to the MCDUs consist of incorporating two new function keys, ATC and FMC COMM. The ATC key allows the crew to access the new ATC data-link page directly. The FMC COMM button allows access to the new AOC data-link page.

The AMU upgrade consists of a revised EPROM to support the new FANS functions in the FMS.

The Multi-input cockpit printer is upgraded and connected to the FMCs, allowing the crew to print out data-link messages such as clearances and route information.

SATELLITE COMMUNICATIONS SYSTEM

The SATCOM system is a communications system that uses satellites as relay stations. The system consists of three components, the satellite, the Aircraft Earth Stations (AES), and the Ground Earth Stations (GES). The satellite relays radio signals between AES and the GES. The AES on the aircraft interfaces with various on-board avionics systems. The GES is a fixed radio station that interfaces with ground based communication networks. The SATCOM network provides data, voice, and passenger communications for aircraft worldwide. The Collins SAT-906 system has been installed on many of United's 747-400 aircraft. This system consists of a Satellite Data Unit (SDU), two High Power Amplifier (HPAs), a Radio Frequency Unit (RFU), a low gain antenna (LGA), a high gain antenna (HGA), and a beam steering unit (BSU). The SDU manages the satellite communication protocols and the interface to all other aircraft systems. The SDU provides up to six channels for cockpit data communications as well as both cockpit and cabin voice communications simultaneously. One channel is always reserved for cockpit data communications to ensure data-link reliability. The SDU is connected to the cockpit MCDUs via an ARINC 739 serial communications interface. The MCDU provides the crew control interface to SATCOM allowing manual access to items such as telephone directories, owner requirements tables, and maintenance functions. Although, three MCDUs are connected to the SDU, the SDU can only be actively controlled by one MCDU at a time. The SDU is also connected to the flight crew's audio control panels (ACP) for the placement of voice calls. The AMU handles the aircraft data-link functions either through VHF radio or SATCOM. Message data is exchanged with the AMU on a low speed ARINC 429 bus using a bit oriented communications protocol specified by ARINC 429.

The SATCOM implementation on the NASA B747-400 simulator is composed of the host based AES simulation, the GES simulation on the EOS, the ARINC 739 interface with the MCDUs, and the ARINC 429 Williamsburg interface with the AMU. The AES model generates all the SATCOM MCDU pages available to the crew and simulates their functionality except for the maintenance functions. The ACP operations and system malfunctions are also handled by this model. The simulation of the ground station is two-fold, comprising of audio control and data-link control via the EOS.

INTEGRATED DISPLAY SYSTEM UPGRADES

The Integrated Display System (IDS) is composed of the Electronic Flight Instrument System (EFIS) and the Engine Indication and Crew Alerting System (EICAS). The EFIS portion of the IDS is composed of the Primary Flight Display (PFD) and the Navigation Display (ND) for both the Captain and First Officer. The PFD displays attitude/direction/altitude/airspeed information, while navigation, performance, and weather radar information is displayed on the ND. The EICAS displays engine data, caution and warning messages, and subsystem data and faults. On the B747-400 simulator, the IDS is programmable to support the development of new or revised flight displays and/or system synoptics for aviation safety research. The actual aircraft IDS is normally comprised of six integrated display units (IDUs), and three Electronic Interface Units (EIUs). On the simulator, the IDUs are simulated through the use of six commercial, raster-based CRT displays which are driven by High Resolution Graphics Cards (HRGCs). When the simulator is loaded, the HRGCs contain down load files which include all the EFIS and EICAS display information. The down load files are programmable in order to create or modify existing IDS display pages. The IDS simulation model, resident on the host computer, provides the EFIS and EICAS system logic to drive, in real-time, the HRGC graphics software on the IDS pages. The software also provides for EIU functions, including data processing and formatting for EICAS message generation, and interfacing with other simulator hardware and software systems.

The addition of two new systems, GPS and SATCOM, and the upgrade of the FMS, introduced the need for specific EICAS messages and EFIS modifications. The new EICAS messages warn the flight crew of GPS sensor and SATCOM faults. Advisories and cautions are also generated for RNP inadequacies and data-link status. Changes to the host computer software were made to incorporate these new EICAS messages. The EFIS was modified to add a unique symbol to indicate the GPS position on the ND. This symbol required the modification of the ND HRGC down-load files and the IDS simulation. For both the EICAS messages and the ND modifications, new FMC outputs had to be accessed and made available to the host software in order to drive the new elements.

Another component of the B747-400 aircraft warning system is the Modular Avionics and Warning Electronics Assembly (MAWEA). The MAWEA handles data/configuration management, fault management, crew control/indication, and guidance. As part of the FANS upgrade, the crew alerting module was modified to activate the low level chime upon receipt of an ATC uplink and an aural alert if the RNP is exceeded in the approach phase. Modifications were also made to the host based MAWEA model and other associated software in order to monitor specific FMC outputs that trigger aural alerts.

Figure 2 shows the various modifications as part of the FANS upgrade on the NASA 747 simulator, depicting changes to the simulated systems and affected aircraft avionics.

SIMULATED GROUND-BASED ENVIRONMENT

The ADS and CPDLC functions are generally hosted on controller workstations with graphical user interfaces for selecting or displaying the data-link information for the airplanes flying in the airspace in the controller's jurisdiction. The simulated environment for the current FANS upgrade is supported via specially designed control pages on the 747-400 simulator's EOS. Each of the FANS data-link applications has a ground-based counterpart that must process each message sent to or received from the data link avionics. Providing the ground support functions constituted the bulk of the

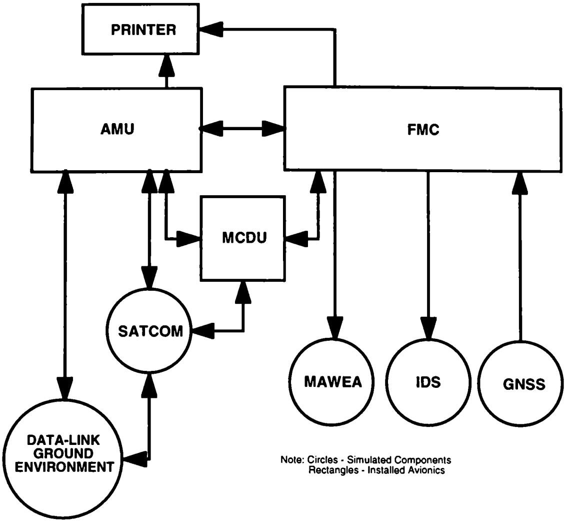

PRINTER
AMU
FMC
MCDU
SATCOM
MAWEA
IDS
GNSS
DATA-LINK GROUND ENVIRONMENT
Note: Circles - Simulated Components
Rectangles - Installed Avionics

Figure 2 - Simulated Systems & Affected Avionics

software development effort for the FANS upgrade. This is also an area, unlike the airborne upgrades, where functionality and level of fidelity had to be designed for a phased development approach commensurate with near term research needs.

EOS GROUND STATION INTERFACE

The EOS is the primary interface from which the simulator set-up and control operations are performed. At the CVSRF, this state of the art system has been enhanced to also serve as the main experimenter control station.

An on-board station located in the aft of the simulator cab, and an off-board station located in the Experiment Lab, each consists of two interactive CRT displays driven by SGI Indigo graphics workstations. The EOS has a customized graphical user interface consisting of input buttons, pop-up menus, display windows, etc. The interface is user friendly and intuitive to use. Display page items are easy to program and utilize a script which is interpreted by a graphics rendering system.

The EOS is a logical choice as a user interface for the ground-based FANS data-link applications. The EOS already has pages to support ground based ACARS functions. In this FANS upgrade, additional pages are defined for FANS data-link applications. All data-link pages including the existing ACARS pages have been consolidated under a Datalink menu accessible from the EOS Main menu.

MESSAGE PROCESSING ARCHITECTURE

As mentioned earlier, ACARS functions commonly used in airline operations are included in the basic simulator capabilities. Simulation support for ground based ACARS functions consists of a set of ACARS message processing modules resident on the simulator's host computer. These modules interface to the AMU via a VHF data link. The ground user interface is provided on the EOS. The uplink messages sent to, and downlink messages received from the AMU, are all in the standard ACARS message format previously discussed.

In the FANS upgrade, a SATCOM data-link interface to AMU has been added. This will provide a data communications link where VHF coverage is not available. Additional software modules have also been added to support the special processing of the ADS, ATC and AOC data-link messages. Figure 3 is a generalized block diagram of the data-link ground based message processing components. The ADS and ATC DL messages are in bit-oriented format and therefore require the ACF to make conversions necessary to ACARS character format. In addition, all three message types include a Cyclical Redundancy Check (CRC) to ensure end-to-end integrity. Messages belonging to the bit-oriented category also utilize special encoding rules to translate message data, obtained from user selections on the appropriate EOS data-link page, into a binary stream. The encoding rules are defined in RTCA DO-212, 219 for the ADS and ATC DL respectively. Each 4-bit nibble in this binary stream is then replaced with the corresponding ISO-5 character as per the ACF. The resulting ACARS compatible characters are then encapsulated in a standard ACARS message format before transmission. These are the typical steps involved in the processing of a bit-oriented uplink message. The steps are reversed in the case of a downlink. Figure 4 shows schematically, the steps involved in a typical ATC DL uplink message processing.

The AOC DL messages utilize character-oriented messages and therefore can skip several of the message processing steps that apply to bit-oriented messages. However, the data in these messages are specified in a special syntax as per ARINC 702. Figure 5 is an example of a typical AOC DL uplink message.

GROUND BASED ADS

The ADS ground user interface page is accessible from the Datalink menu under the EOS main page. Three different contracts and their associated report processing are currently supported. The following are some specifics relating to the ADS implementation:

- Periodic - ADS Basic, Fixed Intent and Event contracts.
- Contracts initiated via dedicated buttons for each type. User selected parameters like the reporting rate, fixed intent time, altitude range, lateral deviation, etc., are input via dialog boxes.
- Display windows for each report.
- Button to terminate all contracts.

GROUND BASED AFN

Three downlink messages and two uplink messages comprising the Initial Notification and Contact Advisory are supported. AFN functions are combined on a single page accessible from the Datalink menu.

- Buttons representing three different ATC oceanic centers are used to acknowledge a contact initiated from the aircraft. A contact received status is indicated by changing the button color to red. On acknowledgment the status changes to green. An auto response button, when selected, automates the Initial Notification transaction by automatically sending an acknowledge response to a Contact downlink.
- Contact Advisory transaction may be initiated by first selecting a separate button provided.

GROUND BASED ATC DL

A considerable development effort was spent providing the ground support functions for ATC DL. As a result, nearly all downlink messages and a majority of the uplink messages comprising the total message set is now functional. The ground user interface is graphical and interactive as in the real world CPDLC workstations. It was designed in a flexible manner to support future enhancements. The ground support functions have been modularized in such a way the interface is separate enough from the message processing architecture so as to be easily modified without affecting the other components. Following are some of the salient features of the ATC DL user interface design:

- Separate pages provided for uplink and downlink operations. Both uplink and downlink pages are accessible from the Data Link menu. In addition uplinks can be reached from the downlink page and vice versa.
- Separate buttons provide access to the various functional message groups for uplink message selection. User selected parameters are input via pop-up dialog boxes. The solicited data values for these inputs are automatically limited to allowed ranges.
- Once the message and all associated data are selected, the composed uplink is shown in a preview window. The controller may then opt to either send or delete this preview buffer.
- A separate window is provided on both the uplink and downlink pages for logging a record of the transacted messages. Each message is shown with the data authority with which the transaction has taken place, time stamp, a unique message identification number, a message reference number and a status of the message. The message reference number is used as a reference number for all response type messages.
- ATC Data authority connect and transfer operations are provided on a separate page accessible from the Data Link menu. Three types of uplinks supporting connecting, ending and assigning a next oceanic center for data service are available.

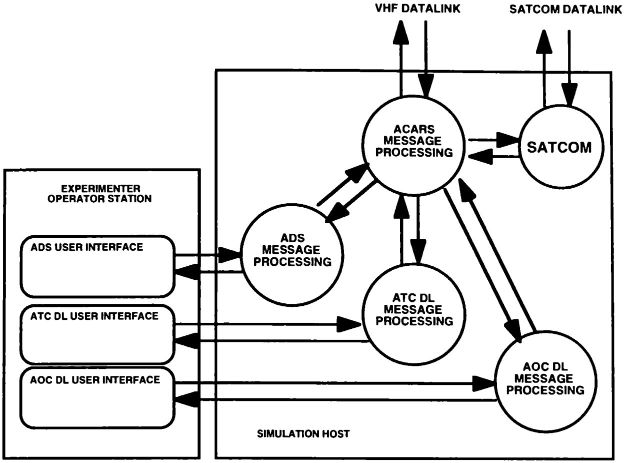


Figure 3 - Simulated Ground Based Data-Link Environment

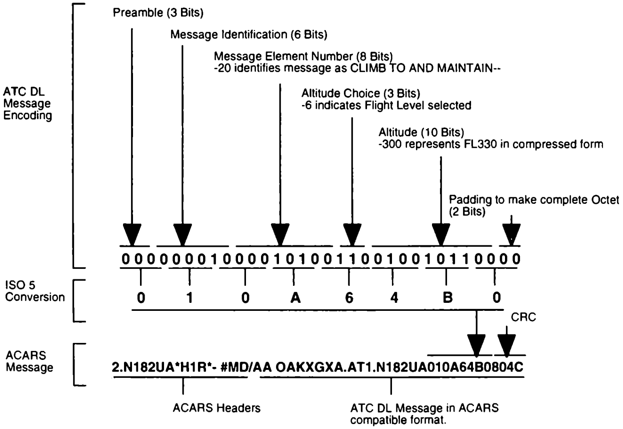

ATC DL Message Encoding
Preamble (3 Bits)
Message Identification (6 Bits)
Message Element Number (8 Bits)
-20 identifies message as CLIMB TO AND MAINTAIN--
Altitude Choice (3 Bits)
-6 indicates Flight Level selected
Altitude (10 Bits)
-300 represents FL330 in compressed form
Padding to make complete Octet
(2 Bits)
00000001000010100110010010110000
ISO 5 Conversion
0 1 0 A 6 4 B 0
CRC
ACARS Message
2.N182UA*H1R*- #MD/AA OAKXGXA.AT1.N182UA010A64B0804C
ACARS Headers
ATC DL Message in ACARS compatible format.

Figure 4 - Example of a typical ATC DL Message

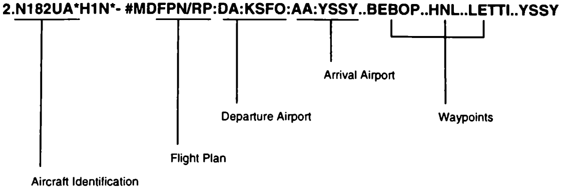


Figure 5 - Example of a typical AOC DL Message

GROUND BASED AOC DL

Flight plan and winds uplinks are currently supported under AOC DL. The ground operations page is accessed via the Datalink menu. Since the AOC DL messages are character based, they can be easily composed using an ASCII text editor using the syntax rules defined in ARINC 702A. For the current implementation, three flight plan uplinks and the associated winds data uplinks have been pre-defined.

- Flight plan and wind requests upon receipt are annunciated on the ground user interface.
- Buttons are provided to select any of the available flight plan or wind messages for uplink.

RESEARCH PROGRAMS

With the introduction of FANS in the NASA 747 simulator, a unique research capability is provided to the aviation community. Research programs dealing with issues such as the utilization of advanced digital information transfer between aircraft and ground-based functions can be examined or evaluated in a full-mission simulation environment. One of these programs already in progress is the Federal Aviation Administration's Dynamic Aircraft Route Planning (DARP) System Program. This program is concerned with the viability of sending rerouted flight plan information to aircraft over the ocean by exploiting the capabilities of emerging satellite-based communications and navigation systems, enhanced avionics capabilities, and advanced ground-based air traffic control systems. The DARP System requires four entities to participate and collaborate. These include the Traffic Management Unit (TMU) specialist, the Airlines Operations Center (AOC) flight dispatcher, the flight crew and air traffic control. In the DARP process the Traffic Management Unit (TMU) specialist is responsible for developing a new route based on the latest weather information. The Airlines Operations Center (AOC) flight dispatcher is responsible for developing a revised flight plan and then transmitting it to the aircraft. The flight crew's role is to review and accept the new flight plan and to request a new route clearance from ATC. ATC's role is to develop, coordinate, and issue the new route clearance. With the CVSRF's implementation of FANS on the 747-400 simulator, the ground-based functions are being emulated to support these air-ground digital communications, making it an ideal research tool for conducting studies regarding DARP operational and human factors issues. Other research programs that will benefit by the implementation of FANS include NASA's Advanced Air Traffic Technology (AATT) Program and the FAA's Free Flight Program. These studies will examine the impact of implementing advanced avionics and ground control systems on the current air traffic management system. Some of these programs will investigate new systems and capabilities that will evolve as a result of FANS. The introduction of new strategic collision detection systems making use of ADS information will also be examined, as well as the development and integration of advanced flight displays depicting enhanced traffic symbology, and reduced separation, vertical situation, and advanced ground taxi displays. Eventually, enhancements to existing GPS capabilities will provide the ability to conduct precision approaches. In addition, terminal area operations exploiting augmented GPS and RNP technologies will be evaluated for guiding aircraft along complex curved approaches.

FUTURE UPGRADES

With the FANS-1 upgrade described in this paper, the CVSRF 747-400 simulator is up to date with the latest upgrade currently available to the airplane. As a simulator certified to the highest recognized standards, it can be expected to keep up with all the forthcoming avionics upgrades relating to the FANS airborne environment. These upgrades in the near term will likely include a revision to the FMC operational software and perhaps new data-link features. Any such upgrade to the installed avionics is easily supported. The ground-based support functions provided in the current upgrade fall in the realm of a single flight simulator environment. In order to support a futuristic CNS/ATM environment, the ADS and ATC DL should be part of a ground-based ATM dealing with a realistic airspace consisting of multiple FANS equipped and unequipped aircraft. This type of ground-based simulation environment currently exists at the FAA Technical Center. Researchers at Ames are also eager to explore FANS technologies, especially as they pertain to terminal area operations. A more extensive FANS ground environment, undertaken as part of the CVSRF's Air Traffic Control simulation facility is entirely justifiable in anticipation of future research.

SUMMARY

There are many benefits to be achieved by the implementation of FANS. These include an increase in safety and efficiency in flight operations, an improvement in service to passengers, and a reduction in airline operating costs. The increase in safety and efficiency improvements in flight operations will be made possible by the emergence of enhanced communications, navigation and surveillance capabilities. The airlines and ATS providers will have full access to the latest weather information, aircraft status and ATC facilities which will enable optimal use of the available resources. The ATS providers will have better planning tools and can sustain optimal routing with fewer delays. The ATM system will also be able to adapt quicker to changing conditions such as weather or system failures. Service improvements to passengers will be based on the increased communications capabilities introduced by data and satellite communications systems. This will introduce the ability to provide on-board reservations, access to various information services and enhanced passenger telephone networks. These benefits are likely to increase the number of passengers and provide additional revenue for the airlines as well. Efficiency improvements in flight operations should yield reductions in airline operating costs, resulting in higher payloads, improved aircraft utilization, reduced flight crew time, reduced maintenance costs and lower fuel costs.

Examples of direct benefits achieved by FANS will include the ability for pilots to automatically load up-linked flight plan changes into the FMC without having to manually type in the revised information. This provides a more efficient means of accepting clearances than what is used today. This will prevent typing errors or miscommunication errors, ensuring greater efficiency and enhanced safety. With the introduction of ADS, and the ability to up-link an optimal flight plan based on the latest weather information, reduced separation of aircraft for FANS equipped aircraft is also possible. With a smaller separation standard, the resulting improvement in airspace capacity will allow improvements in operating efficiencies for aircraft by reducing the required enroute fuel burn and contingency fuel. This will provide increased efficiency and savings in fuel costs for air carriers. In addition, ADS will enable ATC to detect the position of aircraft more accurately and allow safer and more efficient management of airspace. These features provided by FANS are the first step in transitioning the current air navigation system to a more efficient and safer air traffic management system. Incorporating FANS into the NASA 747-400 simulator will enable NASA to examine human factors and airspace operational issues regarding the automatic information transfer of flight plan changes or other ATC information, not just in the oceanic arena, but eventually in the terminal area as well, providing a safer overall flying environment.

ACKNOWLEDGMENTS

The authors of this paper would like to express their gratitude to the many organizations and people who helped contribute to implementing FANS in the NASA 747-400 simulator. Sincere thanks go to CAE Electronics Ltd., United Airlines, the FAA Technical Center, Systems Resources Corp., Honeywell, and Rockwell-Collins. A special word of thanks go to the many people who supported the FANS project with the numerous hours of effort they put in to make it a success. These include members of the NSI Technology Services Corp. staff, and the respective Branch Chiefs for the Simulation Systems and Simulation Operations Branches, Bob Shiner and Tom Alderete at NASA Ames Research Center, who remained loyal advocates throughout this project.

LIST OF ACRONYMS

AATT	Advanced Air Traffic Technologies
ACARS	ARINC Communications, Addressing and Reporting System
ACAS	Airborne Collision Avoidance System
ACF	ACARS Convergence Function
ACMS	Aircraft Condition Monitoring System
ACP	Audio Control Panel
ADNS	ARINC Data Network Service
ADS	Automatic Dependent Surveillance
AES	Aircraft Earth Station
AFN	ATS Facilities Notification
AFTN	Aeronautical Fixed Telecommunication Network
AMU	ACARS Management Unit
ANP	Actual Navigation Performance
AOC	Airline Operations (Operational) Communications
AOC DL	AOC Data-Link
ARINC	Aeronautical Radio Inc.
ASCII	American Standards Committee for Information Interchange
ASM	Air Space Management
ATC	Air Traffic Control
ATC DL	ATC Data-Link
ATCRBS	ATC Radar Beacon System
ATFM	Air Traffic Flow Management
ATM	Air Traffic Management
ATN	Aeronautical Telecommunications Network
ATS	Air Traffic Services
BSU	Beam Steering Unit
CIDIN	Common ICAO Data Interchange Network
CPDLC	Controller Pilot Data-Link Communications
CNS	Communications, Navigation, Surveillance
CRC	Cyclical Redundancy Check
CRT	Cathode Ray Tube
CVSRF	Crew-Vehicle Systems Research Facility
DARP	Dynamic Aircraft Route Planning
DME	Distance Measurement Equipment
DOP	Dilution-of-Precision
EFIS	Electronic Flight Instrument System
EICAS	Engine Indication and Crew Alerting System
EIU	Electronic Interface Unit
EOS	Experimenter Operator Station
EPROM	Erasable Programmable Read Only Memory
FAA	Federal Aviation Administration
FANS	Future Air Navigation System
FIR	Flight Information Region
FMC	Flight Management Computer
FMS	Flight Management System
GES	Ground Earth Station
GPS	Global Positioning System
GLONASS	Russian Federation Global Navigating Orbiting Satellite System
GNSS	Global Navigation Satellite System
GNSSU	Global Navigation Satellite Sensor Unit
HF	High Frequency
HGA	High Gain Antenna
HRGC	High Resolution Graphics Card
HPA	High Power Amplifier
IATA	International Air Transport Association
ICAO	International Civil Aviation Organization
IDS	Integrated Display System
IDU	Integrated Display Unit
IRS	Inertial Reference System
ISO	International Standards Organization
LGA	Low Gain Antenna
LRU	Line Replaceable Unit
MAWEA	Modular Avionics and Warning Electronics Assembly
MCDU	Multifunction Control Display Unit
NASA	National Aeronautics and Space Administration
ND	Navigation Display
OOOI	Out-Off-On-In
OSI	Open Systems Interconnect
PFD	Primary Flight Display
RAIM	Receiver Autonomous Integrity Monitoring
RFU	Radio Frequency Unit
RISC	Reduced Instruction Set Computer
RNP	Required Navigation Performance
RTA	Required Time of Arrival
SATCOM	Satellite Communications System
SDU	Satellite Data Unit
SGI	Silicon Graphics Inc.
SITA	Societe Internationale de Telecommunications Aeronautiques
SSR	Secondary Surveillance Radar
TCAS	Traffic Alert and Collision Avoidance System
TMU	Traffic Management Unit
URE	User Range Error
UTC	Universal Time Clock
VHF	Very High Frequency

REFERENCES

1. FANS CNS/ATM Starter Kit, International Air Transport Association, International Civil Aviation Organization, Montreal, Quebec, Canada, 1995.

2. Aleshire, W., United Airlines B747/400 FANS One Flight Management System, February 1995.

3. System Description for the HG2021 Global Navigation Satellite Sensor Unit (GNSSU) for Air Transport Systems Division, CFS-MO Honeywell Inc., May 1994.

4. Instruction Guide - What is SATCOM?, 523-0776032, Collins Air Transport Division, Rockwell International, January 1990.

5. CAE Boeing 747-400 Full Flight Simulator, Software Documentation, Avionics - ACARS, 10014032 Rev. 0, CAE Electronics Ltd., St. Laurent, Canada, February 1994.

6. CAE Boeing 747-400 Full Flight Simulator, Software Documentation, Software Avionics, Integrated System Display, 10015347 Rev. 0, CAE Electronics Ltd., St. Laurent, Canada, January 1994.

7. Sullivan, B., Soukup, P., The NASA 747-400 Flight Simulator: A National Resource for Aviation Safety Research. AIAA Flight Simulation Technologies Conference, San Diego, California, July, 1996.

8. Kelly, R. J., Davis, J. M., Required Navigation Performance (RNP) for Precision Approach and Landing with GNSS Application, Navigation: Journal of the Institute of Navigation, Vol. 41, No. 1, Spring 1994.

9. ARINC 618-1, Air-Ground Character-Oriented Protocol Specification, Aeronautical Radio Inc., December 1994.

10. ARINC 620-1, Data Link Ground System Standard and Interface Specification (DGSS/IS), Aeronautical Radio Inc., January 1994.

11. ARINC 622-2, ATS Data Link Applications over ACARS Air-Ground Network, Aeronautical Radio Inc., January 1993.

12. ARINC 702A, Flight Management Computer System, Aeronautical Radio Inc., August 1995.

13. ARINC 724B-2, Aircraft Communications Addressing and Reporting System (ACARS), Aeronautical Radio Inc., November 1993.

14. RTCA DO-212 , Minimum Operational Performance Standards for Airborne Automatic Dependent Surveillance (ADS) Equipment, RTCA Inc., October 1992.

15. RTCA DO-219 , Minimum Operational Performance Standards for ATC Two-Way Data Link Communications, RTCA Inc., August 1993.

AIAA-96-3522-CP

SIMULATION TOOLS WITHIN AN AIRCRAFT UPGRADE PROGRAM - A COST EFFECTIVE APPROACH

Dr. Yosef Gindin
Eliezer Darom
Elbit Defense Systems
Elbit Ltd.
Haifa, Israel

1. ABSTRACT

The modern approach to aircraft upgrade, and military aircraft in particular, is the integration of an advanced avionics system into an existing aircraft.

An avionics system development program involves several stages, requiring extensive simulation activities. These activities are performed using a set of simulation tools, which support the different tasks.

Elbit Ltd. is a leading company in avionics systems development and military aircraft upgrade programs. During the development of simulation tools for these programs, we have come up with a cost-effective solution, which enables us to implement all the simulation tools utilizing a modular H/W architecture and a single software development effort. The simulation stations are built of a standard H/W core, to which different modules are connected, thus creating the specific stations.

Elbit's modular design provides easy duplication, modification, expandability, interchangeability and re-use of the simulation equipment. It reduces the overall risk, cost and schedule of the upgrade program.

This paper describes the simulation processes and tools used during the various development stages, as well as a detailed description of Elbit's simulation concept and tools.

2. INTRODUCTION

The modern approach to aircraft upgrade, and military aircraft in particular, is the integration of an advanced avionics system into an existing aircraft. This solution gives a cost-effective improvement of the aircraft operational capabilities, while keeping the existing airframe. An upgrade program provides the aircraft with new operational capabilities, resulting from expanded computation power, accurate sensors and graphical pilot interface.
Aircraft upgrade means integration of several new systems into the aircraft, which interface and transfer signals and data between each other and with the existing aircraft systems.

The primary benefit of aircraft upgrade, compared to new aircraft purchase, is the reduced cost. An upgrade program results in a state-of-the-art weapon system, at a cost much less per aircraft, and eliminates the logistic effort of adding a new aircraft. In order to reduce the upgrade cost it is important that the simulation tools, as well, are cost-effective relative to the whole program cost.

2.1 ELBIT UPGRADE APPROACH

Elbit's upgrade approach is based on a central Mission Computer (MC), which interfaces with the various sensors, pilot cockpit interface units and other aircraft systems. This single computer supports most of the functions which were distributed amongst several different computers in earlier systems, namely Mission computations, Armament system management, Display generation, Communication and Radio Navigation, Helmet Sight and more.
This approach implies one core simulation tool, which simulates the full environment of the MC. This simulation tool provides the MC with operational data and signals, based on accurate simulation models, via its real H/W interface. A sub-set of the simulation tool simulates external input signals to the real avionic sensors. When the real sensors are connected, during system integration phase, the whole avionics systems operates as in a real flight.

Elbit's simulation tools have been used in several upgrade programs - F5, MIG-21 and others - in several countries. The tools, which were developed during the first upgrade program, are being used in the new programs with only minor changes (required due to different H/W configuration or aircraft type).

3. SIMULATION WITHIN AN UPGRADE PROGRAM

3.1 SIMULATION TASKS

In an upgrade program there are several tasks, requiring intensive usage of simulation tools. These tasks are executed by various personnel, with different requirements for models definitions, fidelity and accuracy. These main tasks are listed below:

1. Hardware Development

The Hardware Development task caters to the MC H/W design, development and testing.

Usually this task deals with raw ICD data, and does not need the information about system architecture, or any operational data. The required simulation for this task is full I/O protocol simulation and monitoring, with low level errors simulation (such as protocol errors, timing errors, noise, etc.)

2. Basic Software Development

The Basic S/W usually includes the operating system, MC timing, I/O drivers, ICD implementation, etc.

The required simulation for this task is similar to the H/W development task, with the addition of relevant data simulation in the serial link channels according to the ICD.

.

3. Application Software Development

The Application Software development task implements the MC main avionics algorithms (such as ballistics, navigation, data processing, pilot MMI implementation, avionics' LRU control).

This task requires the most extensive use of the simulation tools and models: the algorithm evaluation and accuracy determination require a set of sophisticated models of the avionic sensors and ownship motion. Additionally, models of the various weapon types (bombs, bullets, missiles, rockets, etc.) should be used for evaluation of the weapon delivery algorithms accuracy.

4. Formal Qualification Tests (FQT)

The FQT activity covers the avionics Operational Flight Program (OFP) formal testing and Proof Of Design (POD).

This task needs the highest level of simulation, in order to prove compliance with the system design and required accuracy. The tests are conducted in "real" flight conditions, which are achieved by the high-fidelity simulation.

Automatic tools for tests definition and activation, such as scenario language and scripts, are strongly recommended for this task in order to enable a semi-automatic process, with exact repetitions of complex test procedures.

5. System Integration and ATP

In the system integration phase the MC is integrated with the real avionic units, and the system is tested as a whole. This is the final testing phase before the flight tests.

The simulation requirements for this task are real time simulation of the aircraft motion, and its natural (weather) and tactical (targets, threats, etc.) environment. The models are activated in real time mode, and interface with the real avionic units (INS, radar, ADC, etc.). I/O monitoring is very important in this phase, in order to identify integration problems between the different units.

The closer the simulation resembles real flight conditions, the more confidence can be achieved before actually flying the system. A thorough integration and testing phase can reduce the amount of test flights, thus cutting the overall program costs.

6. Flight Test Support

The simulation tools are used for reproducing certain flight conditions in order to identify and diagnose problems which appeared during flight. This enables solving the problem by using ground tests rather than airborne tests, and again reduce the flight tests cost.

Those tests are performed on one of the development stations, rather than on the flight test instrumentation.

7. Pilot Training

An Avionics Trainer (AT) is a necessary part of the upgrade program. The customer will demand a training device for the new avionics system.

The best solution for 1-2 Avionics Trainer units is a trainer design, based on the real MC. It is meant to train pilots, most of whom are already familiar with flying the aircraft, in operating the new avionics suite in the air. The AT incorporates a real MC, and real pilot MMI units, in a mock-up cockpit. The simulator S/W simulates inputs to the MC, and receives input from the pilot controls.

Training in the AT gives the pilot experience in operating the real avionics system, and yet saving costly flight hours.

8. Mission Planning

A Mission Planning system (MPS) enables the pilot to plan his missions on a graphic workstation, and calculate all the required parameters for the flight.

The simulation S/W of the MPS performs two tasks:

- Calculation of aircraft performance - time, fuel consumption, maneuvers - based on aircraft configuration, weather conditions and flight plan.
- Dynamic calculation producing a 3 dimensional (3D) mission rehearsal. This simulation task is performed together with 3D imaging of the terrain along the flight route, and enables the pilot to "fly" his mission without leaving the squadron.

9. Mission Debriefing

The ACMI (Air Combat Maneuvering Instrumentation) is a debriefing system which enables graphic presentation of an air-to-air combat maneuvers by integrating accurate flight data from several aircraft.

Although this is usually an independent task, part of the simulation models are used in this project. A good example is the weapons models: during the exercise flight , the simulated weapon release commands are recorded, and on the debrief station, after flight, the weapon trajectory is simulated and hit or miss case is decided.

10. Hardware Maintenance

Maintenance equipment is provided to support LRU maintenance for the life-time of the avionics system.

The LRU is tested by activating it, simulating inputs and monitoring the outputs. The simulation and monitoring parts of the LRU test make use of the H/W engineering and Basic S/W simulation tools and procedures.

3.2 MODELS

The simulation task is comprised of two main elements:

One - simulation of the I/O protocol of the simulated unit. This is implemented by H/W and basic S/W.

Two - simulation of the unit functional and logic behavior. This part is implemented in way of modules: a different module for every simulated LRU, and some times two models of the same unit, providing different levels of simulation accuracy and functionality.

The simulation models may be subdivided into the following classes :

1. Avionic Units (LRU) Models

The LRU models describe the behavior of the avionic sensors, weapon system and pilot's MMI devices which are used in the project. There are four main LRU models groups:

- Avionics system sensors - INS, GPS, Radar, RALT, ADC, Radio Navigation (R-NAV) sensors (DME, VOR, ILS & Tacan receivers), RWR.
- Cockpit pilot's MMI devices : Multi-Functional Displays (MFDs), control panels, Head Up Display (HUD), Helmet Mounted Display, flight controls (stick and throttle), HOTAS switches.
- Stores and EW models : Stores, ECM, chaff and flare pods system, etc.
- Communication units: Radios, IFF, ATC, Tacan.

There are different levels of LRU models, all of which are required:

a. I/O model - provides the interface simulations, data injection/monitoring in engineering and raw forms, protocol error injection, etc.

b. Logic level - level A + logic of operation implementation, all modes simulation, modes transfer, power up processes simulation, BIT (built-in-test) responses, etc.

c. Full model : level B + link with motion and environment models, simulation of the LRUs behavior, modeling of the LRU physical errors (a good example is INS Shuler error simulation).

2. Ownship Motion Models

The Ownship Motion Models provide dynamic simulation of the ownship (aircraft or helicopter). The output data of the models are the input data for the high level LRU simulation.

There are two types of ownship model:

a. Full non-linear 6-DoF motion model, based on the aerodynamics, engine, weight and geometric characteristics of the modeled aircraft. This model is tested and verified relative to aircraft flight characteristics documentation.

The model main capabilities are:

- Flight in the full range of the A/C operating limitations
- Atmosphere simulation for standard, hot, cold, polar and tropical types.
- Atmosphere distribution model - constant wind, gusts, turbulence, wind shear.
- Engine simulation including thrust, fuel flow calculation, and time transients.
- Various types of Earth models.
- Control of aircraft : manual, route flight, flight according to autopilot commands

b. Simplified 3-DoF motion model. This is a generic model which provides a simple motion simulation, and is independent of the aircraft type.
   The model main capabilities are :
   - Flight in the full range of the A/C operating limitations
   - Atmosphere simulation for standard day with pre-defined temperature deviation
   - Horizontal, constant wind simulation
   - Control of aircraft: Selection of maneuvers: turn, pull-up, to steerpoint, with roll 0, random and manual flight.

3. Tactical Environment Models

These models generate input data for avionic sensors models, such as Radar, RWR, etc. The models from this group include the following types:

- Airborne Targets model - a simple motion model which provides initial state vector definition and run time simulation of the following maneuvers : pull-up with pre-defined 'g', turn with pre-defined roll or 'g', "follow me", climbing-descent-horizontal flight with pre-defined pitch and heading, flight according to pre-defined route, manual flight.
- Ground Targets model - a simple motion model which provides initial position and velocity definition, and run time simulation of turn or "direct" motion.
- RWR threats.

4. R-NAV Beacons Models

The R-NAV models provide input data for the Radio Navigation sensors models, including VOR, DME, TACAN and ILS beacons models.

5. Weapon Models

The weapon models provide simulation of all types of weapon which will be used by the upgraded aircraft.
The models consist of two parts:
The first part describes the weapon behavior when it is mounted on the aircraft (for example - missile slaving , calculation and logic).
The second part describes the weapon motion after its release. This part provides the integration of motion equations with specific weapon weight, geometric and aerodynamics parameters. (For missiles the seeker and flight control system simulation are provided as well.)
The following weapon models are implemented: A/A missiles, A/G missiles, gun bullets, various types of bombs, and rockets.

The simulation models and their hierarchy in the simulation process are shown in Figure 1 below.

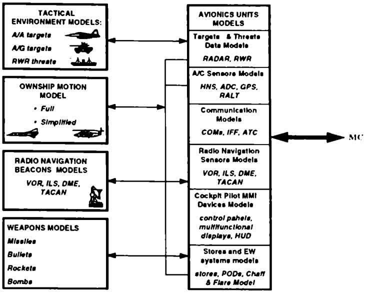

TACTICAL ENVIRONMENT MODELS:
A/A targets
A/G targets
RWR threats
OWNSHIP MOTION MODEL
• Full
• Simplified
RADIO NAVIGATION BEACONS MODELS
VOR, ILS, DME, TACAN
WEAPONS MODELS
Missiles
Bullets
Rockets
Bombs
AVIONICS UNITS MODELS
Targets & Threats Data Models
RADAR, RWR
A/C Sensors Models
HNS, ADC, GPS, RALT
Communication Models
COMs, IFF, ATC
Radio Navigation Sensors Models
VOR, ILS, DME, TACAN
Cockpit Pilot MMI Devices Models
control panels, multifunctional displays, HUD
Stores and EW systems models
stores, PODs, Chaff & Flare Model
MC

Figure 1: Simulation Models and their Hierarchy

It is clear, that different tasks require different models and different levels of simulation. A programmer who implements the CCIP bombing algorithm does not need, and is not interested in, the simulation of the INS checksum error, which is very important for the hardware engineer.

Table 1 below shows the usage of simulation models by the various project tasks.

Task	Avionic LRU models			Ownship		Tactical Environment	R-NAV Beacons	Weapons
	I/O+ICD	Logic	Full	Simple	Full			
H/W	Yes	Partial	No	No	No	No	No	Partial
S/W Basic	Yes	Yes	No	No	No	No	No	Partial
S/W Application	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes
S/W FQT	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
System Integration and ATP	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Avionics Trainer	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes
Mission Planning	No	No	No	No	Yes	No	Yes	Yes
Mission Debriefing	No	No	No	No	Yes	No	No	Yes

Table 1: Use of Simulation Models by Different Upgrade Program Tasks

3.3 SIMULATION ENVIRONMENT

The models which were described above should be controlled, activated, monitored and synchronized with the avionic units and with each other. Each of them needs the environment for model running, interface to the user, to the I/O system and between the models themselves. A set of S/W tools, supporting these tasks, form the simulation environment.

The tools which comprise the simulation environment are described below:

1. I/O Drivers
Provide interface to the I/O H/W to enable real equipment connection.
They comprise of the following drivers:

- Serial link - RS-422, RS-232 and others
- Mux-Bus (MIL-STD-1553)
- ARINC communication channels
- Analog signals
- Discretes signals
- Synchro signals
- Special channels and signals

2. ICD Data Base and Automatic ICD Conversion Tools
A heavy task in every project is the definition, management, and support of the Interface Control Document (ICD). This document defines all data elements and H/W signals in the avionics system, which add up to thousands of elements.
Every data going from or to models should be transferred through an ICD converter, in order to convert the data from the standard representation in the simulation S/W to avionics system format - H/W and S/W. Every task in the project has a very strong interaction with the ICD.

3. Models Scheduler-Dispatcher
Organizes the simulation frame according to models frequency and ICD blocks update rate. The scheduler has a capability for real-time and non real time modes. The basic simulation frame is 20 milliseconds.

4. Simulation Models
As described above.

5. Data Monitoring Tools
Provide the capability for avionics system signals monitoring.
Include the tools for selection of elements to be monitored, data display, storage and replay, and interface to the ICD data base.

6. Universal Data & Error Injection (DEI) Mechanism
Provides data injection capability for every data element in the system. A single MMI function is provided for all the stations. The DEI tool interfaces with the ICD data base.

DEI includes tools for the following tasks:

- Data definition - normal, extreme, out of bounds.
- Error injection - in data level and protocol level.
- Noise injection - level, pattern and frequency.
- Storage of the data exchange process - enables accurate repetition of the executed test.

7. Scripts

This is a tool which supports test automation.

The script tool enables the preparation of a pre-defined script, which controls the events during the specific test according to time or upon event.

Saving the script enables performing the test automatically, and repeating it accurately as many times as required.

Table 2 below describes the use of each simulation tool by the different tasks (use of the models is discussed above).

TOOL	Hardware	Basic S/W	Algorithm S/W	FQT	ATP	Avionics Trainer	Mission Planning	Mission Debriefing
I/O drivers	Yes	Yes	Yes	Yes	Yes	Yes	Partial	Partial
ICD converter	Yes	Yes	Yes	Yes	Yes	Yes	Partial	Yes
Scheduler Dispatcher	Yes	Yes	Yes	Yes	Yes	Yes	No	No
Data Monitoring tools	Yes	Yes	Partial	Partial	Yes	No	No	No
Universal Data & Error Injection (DEI)	Yes	Yes	Yes	Yes	Yes	No	Partial	No
Scripts	No	Partial	Yes	Yes	Yes	Partial	No	No

Table 2: Use of Simulation Tools by Different Upgrade Program Tasks

4. ELBIT DESIGN APPROACH

4.1 SIMULATION IMPLEMENTATION

Simulation support systems have been developed at Elbit and employed in several upgrade programs. The main guidelines for developing these tools were:

- Simulate the full environment surrounding the MC.
- Complete simulation and monitoring capability for every signal and data element in the system. Comply with system timing requirements - computation cycles, transmission rates, etc.
- Single S/W effort to produce simulation models with different levels of accuracy and complexity. The same S/W tools are to be used in different projects with a relatively small adaptation effort.
- Modular H/W design, enabling manufacturing of different support stations using the same building blocks.

  H/W modularity and commonality enable using one station for several tasks, or for different projects, by applying only minor changes.
- Use standard, commercial H/W components as much as possible.
- Focus on Human Engineering criteria to improve the user MMI and operating environment.

4.2 S/W ENVIRONMENT

The simulation tools run on PC (MSDOS and MS-Windows) and Silicon Graphics (UNIX) environment. The usual approach is to use the IBM PC computer for I/O operations, real time ICD conversions, models scheduler, low level LRU models and simple motion models. The complex models, such as full aircraft model, are implemented on the Silicon Graphics computer. The MMI functions, including DEI and simulation control, are run on IBM PC under MS-Windows.

For simple projects and non real time application, the simulation may be implemented on one PC computer. In this case, the scheduler, models and ICD converter are run under DOS windows.

Interface between the computers is implemented by VME shared memory or by network communication.
All S/W is written in C language.

4.3 H/W ENVIRONMENT

The simulation stations are built according to a standard and modular H/W and mechanical design.

The I/O Simulation Hardware (called I/O Machine - IOM), which is the heart of the H/W simulation, is based on VME architecture. Most of the cards used are commercial ones, but some have been developed by Elbit for performing special tasks, such as special communication channels or better performance than the available commercial cards. The IOM design is standard for all simulation stations.

The simulation stations are built by adding different modules - LRU connections, simulation computers, etc. The connection of the different LRUs is implemented via a sophisticated inter-connect panel. The selection between a real and simulated LRU is done by switching a single connector upside-down. For several LRUs, especially MMI units, S/W switching is also provided.

The modular and standard H/W and mechanical design yields the following benefits:

- Similarity between the different stations.
- Easy reproduction.
- Easy modification from task to task and from program to program.
- Reliability.
- Expandability
- Standard and ease of maintenance.

4.4 SIMULATION STATIONS IN AN UPGRADE PROGRAM

The various phases of an upgrade program require several different development and operational stations, each of which utilizes some of the simulation processes which were discussed above.

The following are typical examples of the simulation stations, which are based on the previously described approach. These stations are used in Elbit's aircraft upgrade programs.

Table 3 below shows the upgrade program phases, and the simulation stations which are used in each one.

Phase/Task	Station/Tool
Development	
• Hardware	MC tester
• Software	SW development station, STU
Testing & Integration	
• FQT	STU, STS, AIS
• ATP	AIS
• Flight Tests	STS, AIS
Operational	
• Mission Planning	MPS
• Mission debrief	ACMI, MPS
• Pilots training	Avionics Trainer
Maintenance	
• LRU level	MC I-Level Tester

Table 3: Simulation stations for Upgrade Phases

1. Software Development Station

This type of station is used during the MC S/W development and unit testing at the host computer level. This station consists of an IBM PC which is connected to the embedded S/W host computer. Interface between them is via shared files which contain raw ICD data. The station supports non real-time interface, which is required when developing a real time application on a host computer. Nevertheless, the same simulation S/W is used also for the real time simulation stations.

The following applications run on the PC under MS-Windows:

- Scheduler (in non real-time mode) in OS window
- Simple ownship model
- Avionics LRU models
- DEI tools
- Scripts
- ICD converter

The low price of such stations, and their reusability for other phases/projects, enables the use of several units for a single project.

2. Software Test Unit (STU)

The STU provides the environment to support development, testing and integration of the avionic MC Operational Flight Program (OFP).

Two key features are implemented by the STU:

- Provide the necessary tools for stimulating the MC at all external interfaces.
- Provide features for collecting and analyzing data, to verify the performance of the MC.

The following list summarizes the STU capabilities.

a. Environment simulation with functional or interface models of each LRU that interfaces with the MC.

b. Simplified ownship motion model.
c. Simulated displays representing in-cockpit panels or providing interfaces for panels imitation.
d. Error and data injection for every data element.
e. Data viewing and evaluation in engineering units, raw form, and/or in a graphic format.
f. Conditional online data recording for post analysis.
g. Post-process data reduction and analysis capability.
h. Time tagging of the data elements so that timing measurements between or of events can be accomplished.
i. Integration with the relevant real or simulated cockpit MMI LRUs, panels and switches.
j. Full MC I/O environment including buses, video lines, discretes, analogs, etc.
k. Synchronized operation with the MC.
l. Provide offline utilities for weapon scoring.
m. Run time scripts implementation.

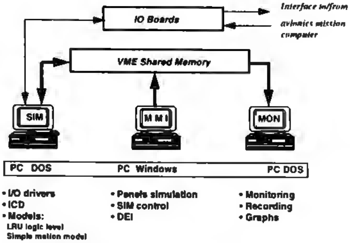


Figure 2: STU Block Diagram

3. Avionics Integration Station (AIS)

The main purpose of the AIS is system integration and ATP. This station has a similar architecture to the STU. The main difference is the H/W and mechanical infrastructure, which enable connection of real avionics LRUs, the same way they are connected in the aircraft. The AIS enables operating the avionic system in its full H/W configuration.

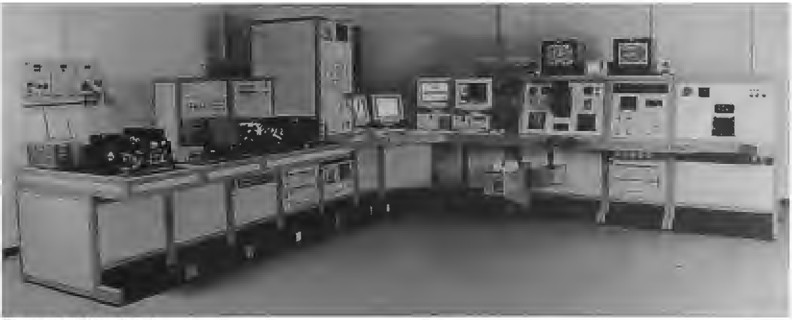


Figure 3: AIS Layout

4. Mission Computer Test Station (MC Tester)

The MC Test Station provides the debug, testing and ATP capabilities for the MC hardware during the development phase. This station consists of two IBM PC computers connected to the IOM:

- Real time main computer - with I/O subsystem interface, ICD converter and low level avionic LRU models.
- MMI, monitoring and DEI computer.

The MC test station can be easily converted into an LRU-level maintenance tester for the MC.

5. Simulation Test Station (STS)

The STS supports the FQT process for the MC S/W. The STS is an upgrade of the STU, with addition of a Silicon Graphics (SG) computer (see block diagram). The main purpose of the SG computer is the implementation of the complex models, which need the high computer power. From the user point of view, the main MMI operations are the same as in the STU.

The following capabilities apply to the STS configuration.

a. High fidelity Flight Ownship Motion model
b. Air and ground targets models
c. High fidelity LRU models

d. STS mission preparation graphical tools

e. view of the simulated sortie tactical scene.

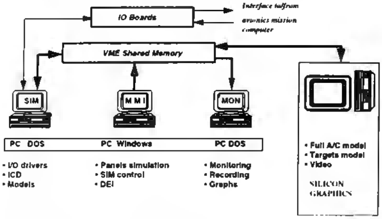


Figure 4: STS Block Diagram

6. Avionics Trainer (AT)

The AT is a necessary part of the avionics systems upgrade program. The AT provides familiarization with the new avionics system, as well as initial and continued training in operating the system during flight.

The AT H/W and S/W are based on the STU configuration, with several additions:

- AT Cockpit, which is a close imitation of the real cockpit, with real avionics MMI devices and operational flight controls.
- Instructor Operation Station, including AT dedicated S/W tasks.
- Out The Window (OTW) display, mixed with real HUD display generated by the MC.
- Avionic sensors video, such as radar, RWR and missiles.
- Audio system.

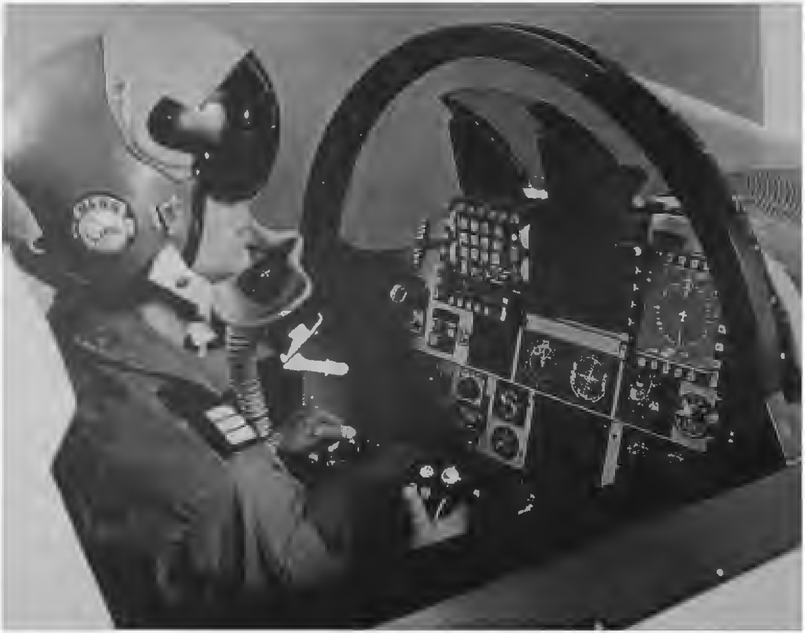


Figure 5: AT Cockpit Layout

The highlights of Elbit's approach to the Avionics Trainer design are:

- Usage of real MMI avionics devices. Avionic switches, panels and displays, which are used by the pilot during the flight, are real (or imitated with high degree of similarity). This is the key for quick familiarization with the new avionics system operation.
- Usage of real avionic MC. It provides the simulator operation with full adequacy to the aircraft avionics system operation.
- Changes to the MC operational S/W are implemented in the AT by simply updating the S/W in the AT MC, without any changes required to the AT dedicated S/W.

The AT design enables to building it with the same architecture and platform as other project test stations. This solution reduces substantially the development and maintenance costs of the AT.

7. Mission Planning System - MPS

Elbit's MPS is a powerful computerized system, which enables the pilot to plan his mission and prepare the flight aids in a precise, convenient, intuitive way, using a graphic workstation. The system provides mission planning, mission rehearsal and mission debriefing capabilities.

The MPS design is based on a UNIX workstation. The S/W is developed using commercial MMI, graphics and database tools.

The MPS simulation tasks are:

- Calculation of aircraft performance - time, fuel consumption, maneuvers.
- Dynamic calculation producing a 3 dimensional (3D) mission rehearsal.

The MPS simulation tasks use the same simulation models which are used in the development stations.

8. ACMI

The ACMI (Air Combat Maneuvering Instrumentation) is a debriefing system which enables graphic presentation of air-to-air combat maneuvers by integrating accurate flight data from several aircraft.
The simulation models used in this station are mostly weapon scoring models. A good example is the A/A missile model: during the exercise flight , the simulated weapon release commands are recorded, and on the debrief station the weapon trajectory is simulated, and hit/miss scoring is decided.

5. CONCLUSIONS

- The avionics system upgrade programs require intensive use of simulation tools, for development as well as for operational purposes.
- All the simulation tools can, and should be, based on the same S/W and H/W architecture, using the 'one time developed' models, with similar user's MMI.
- The described set of simulation tools provides the design for a wide range of test benches and systems, from hardware tester to Avionics Trainer.
- The hi-fidelity simulation, and the surrounding simulation environment tools (I/O monitoring, recording, DEI, etc.) reduce the development phase costs in several aspects:
    - * Accurate, semi-automatic and repeatable test procedure.
    - * 'Purifying' the system to a high level of confidence before the flight tests.
    - * Reproducing flight tests problems on the ground.
- Elbit's solution promotes the program over-all cost effectiveness, by decreasing the price, risk and schedule of the simulation support equipment development and production.

6. ACKNOWLEDGMENT

The concept described in this paper has been designed, developed and implemented by Elbit Test and Support Equipment Engineering Department staff. The Authors wish to thank all those who have contributed their skills, effort and endless days to this task, and made this resulting paper possible.

AIAA-96-3523-CP

INTEGRATED SOFTWARE TOOLS FOR SIMULATION UPDATE AND VALIDATION

Dennis J. Linse[*], Omeed Alaverdi[*], Alexander Kokolios[†], and Michael Anderson[‡]
Science Applications International Corporation, California, MD 20619-3272 USA

Abstract

Developing, updating, and validating simulation models from flight test data are difficult and time-consuming tasks. Much of the time is spent manipulating data and models to the format needed by disparate programs. The Integrated Data Evaluation and Analysis System (IDEAS) gathers existing software into a distributed client/server architecture that integrates all of the tools required to process flight test data and create updated, validated simulation models. By employing uniform data handling routines, IDEAS eliminates unnecessary data manipulation and speeds the overall process. Individual tools require little overhead to access flight data and perform the desired processing. Using modern, networked communications scheme, the system can distribute work among heterogeneous processors while maintaining an integrated working environment. An example of kinematic consistency analysis and one of aerodynamic model correction are provided to show the basic operation of IDEAS in the simulation update and validation arena.

Introduction

Simulation update (improving a math model based on flight test data) and validation (matching a simulation to flight test data) are costly and time consuming tasks. System identification, the rigorous process of deriving a math model from flight test data, is a complicated, multi-step process requiring considerable expertise to perform. In a traditional flight test program, the elapsed time from flying the aircraft and gathering data until the simulation is updated can be months or even years. The primary reasons behind the long times between flight test and simulation update are the labor-intensive processes of data processing and system identification. The process is illustrated in Fig. 1. First, the data must be converted to each of the formats needed by different preprocessing or identification programs. The data are filtered, smoothed, checked for kinematic consistency, and calibrated. The model structure is determined and the model parameters are identified iteratively with intermediate updates compared to the flight data before further computations are made. Finally, the new model must be installed into the simulation and validated against independent flight data. At any point, the procedure may have to be rerun due to poor or insufficient data. Each of these steps may be a different algorithm that requires input data in a distinct format and produces output data in yet another format.

There are nearly as many computer programs to perform the data preprocessing, system identification,

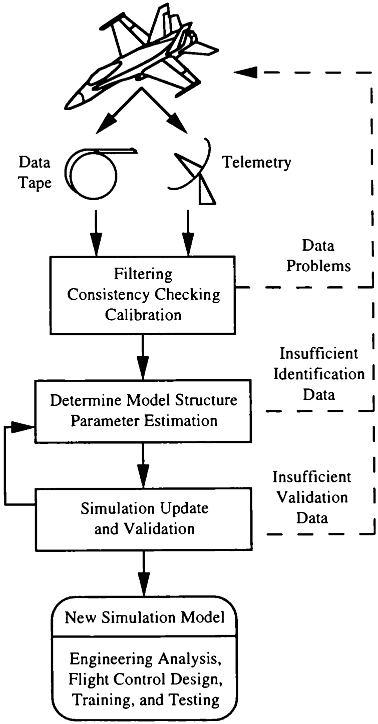

Data Tape
Telemetry
Data Problems
Filtering
Consistency Checking
Calibration
Insufficient Identification Data
Determine Model Structure
Parameter Estimation
Insufficient Validation Data
Simulation Update and Validation
New Simulation Model
Engineering Analysis, Flight Control Design, Training, and Testing

Figure 1. The Simulation Model Update Process.

[*] Aerospace Engineer, Senior Member AIAA
[†] Aerospace Engineer, Member AIAA
[‡] Programmer Analyst

and simulation update process as there are people performing these operations. In general each organization has developed in-house methods that work well for the projects at hand. Some methods have achieved more wide-spread use including MMLE3[1,2] by Maine and Iliff, one of the most well-known maximum likelihood estimation algorithms for aircraft identification; SMACK[3], a smoothing and data consistency checking program; and CIFER[4,5], an integrated system for frequency response identification.

At SAIC, a number of advanced system identification and data analysis methods have been developed. These include Athena[6], an equation error identification algorithm using principal components regression analysis; NAVIDNT[7], a filter error identification programming running a navigation equations algorithm for flight data filtering and data consistency checking; and SCIDNT[8], an output error identification program for nonlinear aerodynamic modeling. These programs have been used individually and jointly with a great deal of success updating aircraft simulations ranging from the F-4[8] to the C-130[9]. Notwithstanding the success of these programs, too much of the overall project time was spent converting to the particular data formats needed by each program and reengineering simulation models from one program to the next.

This paper describes the Integrated Data Evaluation and Analysis System (IDEAS), a complete package to analyze flight test data from end to end. All of the tools needed for flight data analysis, system identification, simulation update, and validation are incorporated into a common architecture. It directly addresses the difficulties of dealing with a multitude of unrelated programs by integrating the tools under a common framework that access a single database of flight data. With all of the components in one place, an expert system shell can be included to direct the identification process and suggest alternatives to the working engineer.

Integrated Data Evaluation and Analysis System

IDEAS is an analysis system designed for manipulation of flight test data emphasizing the creation and validation of complex nonlinear simulations from test data. Its architecture is designed for efficient flight test data archiving and annotation, data preprocessing, consistency analysis and calibration, system identification, simulation update, and validation. IDEAS gathers the many processing steps required to analyze test data into sets of tools controlled with simple commands, windows, and script files. These tools are specifically designed to process flight test data and provide most of the required functionality for any dynamic system test data processing. Since flight testing produces vast amounts of data, communication and database management must still be optimized to handle the volume of data in a reasonable amount of time.

System Processes

IDEAS is composed of a small set of system processes and any number of user processes and tools (Fig. 2). The system processes (shown in gray) that control the basic operation of IDEAS include

- User Interface: textual or graphic
- Database Manager: stores and retrieves all flight and calculated data
- Plot Interface: generic plotting capability
- Error Logger: localized error message support
- Data Interface: conversion into and out of IDEAS format

The backbone of the IDEAS client/server architecture is the *Information Gateway*. The information gateway is a central routing mechanism that makes connections between all processes as required. As each process comes on-line, it registers itself with the information gateway and makes its services available to all of the other processes.

To increase flexibility and expandability, each tool in IDEAS is a separate process. The communication between the distributed processes is conducted using the Data Transfer Mechanism (DTM) developed at the National Center for Supercomputer Applications (NCSA)[10]. DTM is a function library that provides uniform networked communication with transparent data type conversion between heterogeneous processors. It has been optimized for large data messages such as those likely to occur when processing flight data. DTM, a stable, off-the-shelf communication system, provides an excellent base on which IDEAS is built.

The database management system is specialized for flight test data analysis. Data from each flight are stored in distinct records with extensive annotation required to identify the aircraft and conditions of the test. This information provides additional keys to access the flight data at a later time. Histories of each recorded channel are stored separately in a standard (time, variable) format. Each data point is time-tagged; this allows accurate representation of data recorded asynchronously. All IDEAS tools are designed to manipulate asynchronous data properly.

All tools are operated through a uniform interface—either a simple command-line interface or a point-and-click graphic interface (currently X windows using an OSF/Motif interface). An ASCII scripting mechanism allows repetitive functions to be stored and executed on

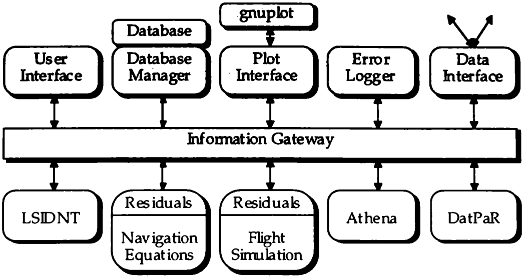


Figure 2. IDEAS and the Information Gateway.

demand. An automatic logging system keeps track of all warnings and errors generated during operation. The error log may be consulted at any time.

Since all flight data within IDEAS are controlled by the database manager process, a new tool must somehow gain access to these data for manipulation. The IDEAS Application Programming Interface (API) provides a generic library of function calls to communicate through the information gateway so that a tool may read and write data or interact with the user. By providing a generic set of data structures designed for flight data along with all of the required input and output routines, the IDEAS API removes all of the drudgery of implementing a new identification tool. The IDEAS API also provides a uniform access method that eliminates the costly process of converting data from one format to another.

In some instances it is desirable to access data stored in the IDEAS database by an external program. For example, MATLAB™ may be used to develop a new processing tool. During initial testing, data from the flight database is converted to MATLAB format by the data interface tool. Likewise data may be converted to and from other common formats including ASCII text files.

To the maximum extent possible, IDEAS was developed using commercial-off-the-shelf or freely distributable software. Maximizing the use of existing software builds on the lessons learned and generates more capable and more robust software. Any new software for IDEAS is designed and coded using standardized languages and operating system constructs. All primary functions are written in ANSI C[11] with access to the operating system through POSIX[12] calls. The IDEAS API provides both a C and FORTRAN interface to the IDEAS database and data manipulation routines.

User Processes

Flight data processing, identification, and simulation update is a multi-step process as indicated in Fig. 1. A number of user processes currently exist under the IDEAS architecture (shown in white in Fig. 2) that address the requirements of this process. Figure 3 shows in more detail how the tools are coordinated to form an integrated identification and simulation update package. Existing IDEAS processes include

- DatPaR: a data preprocessor
- Athena: an equation error identification routine
- LSIDNT: a robust, nonlinear least squares optimization routine
- NAVEQNS: a specialized simulation of the navigation equations of an aircraft
- Simulation Interface: a generic interface to existing aircraft simulations

The DatPaR process (Data Preprocessing and Reconstruction) provides a number of small data processing tools including smoothers, wild-point editors, axis translation and rotation routines, coefficient reconstruction algorithms, and general mathematical functions. Using script files, extensive preprocessing of flight data is accomplished using DatPaR.

An equation error identification algorithm using principal components regression is contained in the Athena process[6]. While a powerful identification algorithm in its own right, Athena is often used interactively for model structure determination. After a model structure is determined and initial estimates of the model parameters are made, an output error identification is used to develop the final parameters.

LSIDNT (Least-Squares Identification) is a general, bounded, nonlinear least squares optimization routine. It is based on the N2F family of algorithms[13]. Splitting the optimization algorithm from the identification tool represents a distinct change from the

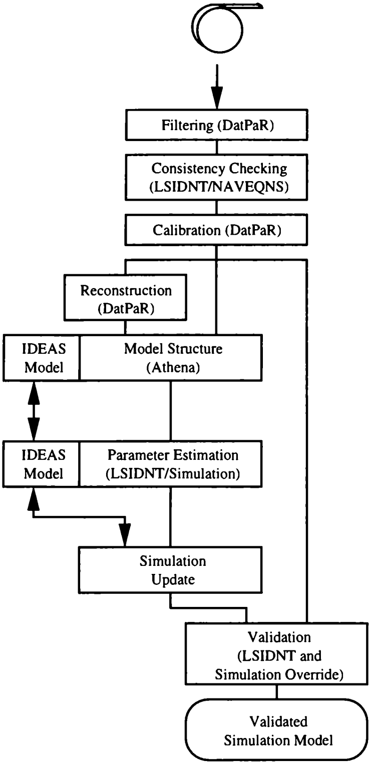

Filtering (DatPaR)
Consistency Checking (LSIDNT/NAVEQNS)
Calibration (DatPaR)
Reconstruction (DatPaR)
IDEAS Model
Model Structure (Athena)
IDEAS Model
Parameter Estimation (LSIDNT/Simulation)
Simulation Update
Validation (LSIDNT and Simulation Override)
Validated Simulation Model

Figure 3. Details of Simulation Update

NAVIDNT[7] and SCIDNT[8] tools. Rather than tightly coupling the optimizer to a particular set of residual equations, it is separated out and used to address several different problems. The single LSIDNT tool is used with different residual calculations in consistency checking, parameter estimation, trim and initialization, and validation.

NAVEQNS is a simulation of the kinematic and navigation equations of a rigid body aircraft flying over a round, rotating earth. Using linear accelerations and angular rates as input elements and sensor measurements as output elements, NAVEQNS is used in conjunction with LSIDNT to form a powerful output error identification tool for sensor calibration and data consistency checking. This combination forms the heart of NAVIDNT.

In addition to the specialized NAVEQNS simulation, a general interface that can be connected to nearly any existing dynamic simulation is included. Any aircraft simulation may be installed by attaching a small executive that coordinates communication between the information gateway and the simulation, controls the simulation operation, and gathers data for residual calculations. When appropriately attached, the simulation is used in conjunction with LSIDNT to form a general, nonlinear, output error identification algorithm. This combination replaces the SCIDNT routines. Two existing aircraft simulations have already been hosted under IDEAS.

IDEAS Model

To further speed up the identification and simulation update process, a parameterized nonlinear model is accessible through the IDEAS API. This model, stored as an easily readable ASCII file, may be used to approximate complex aerodynamic, engine, gear, and other functions. The IDEAS model is constructed with polynomial functions of the flight data, tables, and polynomial basis splines, all of which depend upon user-defined parameters. In most instances, it is used as an incremental model to augment and correct the existing simulation values. For example, Fig. 4 shows how an incremental model is used to implement a lift coefficient correction term. Incremental models are especially useful during a simulation update as the effect of the increment can be easily switched off by setting the corresponding term to zero. This returns the simulation to the prior model.

By using a general model format accessed through a simple API, each identification algorithm uses the same model so that multiple algorithms can quickly be employed to identify a process. For example, an equation error algorithm could be used for quick model structure determination with further refinement of the model parameters coming from an output error process.

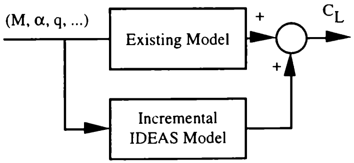


Figure 4. IDEAS Model as an Incremental Model

Extensions

With all of the tools needed for data processing, system identification, and simulation update accessible through a single, scriptable interface, a simulation update expert system that acts as an adviser to the engineer during data analysis and model identification can be added. Gathering knowledge from human experts, the expert system will suggest, direct, and control the overall process. By monitoring the data stream and user actions, the expert system can suggest model structures and tools to the user, advise the user when the data is insufficient to perform the required task, or perform a stand-alone identification using available information. The expert system is useful as a training aid for inexperienced engineers as well as an oversight tool during larger projects. Automating large portions of repetitive tasks is possible as well; the expert system directs and monitors the proceedings providing useful evaluations and comments.

At present, a simple data monitoring expert system using the C Language Integrated Processing System (CLIPS)[14] environment is integrated within the IDEAS environment.

Application of IDEAS

The two examples provided below show the capabilities of IDEAS for simulation update process. The first example demonstrates the kinematic consistency checking and data calibration processes. The second example demonstrates the simulator model update and evaluation capabilities.

The necessary first step to full model identification is calibrating the collected data to ensure kinematic consistency. Starting from flight test data stored in the IDEAS database, the LSIDNT and NAVEQNS tools are employed to estimate scale factors and biases on the measured data. The corrections are used to make all of the measurements kinematically consistent, that is, measured accelerations should integrate to measured velocities:

$$\left(a_x, a_y, a_z\right) \rightarrow (V, \alpha, \beta) \qquad (1)$$

and measured angular rates should integrate to measured angular positions:

$$(p, q, r) \rightarrow (\phi, \theta, \psi) \qquad (2)$$

NAVEQNS performs the integration of these navigation equations. LSIDNT, the least-squares optimizer, iteratively calls the NAVEQNS simulation and adjusts the scale factors and biases on the measured input and output elements to achieve a set that is kinematically consistent.

For the present example, a simulation of the Navion[15,16] is used to generate the simulated flight test data. A 30 second maneuver consisting of a lateral stick doublet followed by a longitudinal stick doublet is performed. The normal acceleration a_z is biased by 2 feet/sec and the angle of attack α is biased by 1 degree and scaled down by 10%. Figures 5 through 8 show the angle of attack, angle of sideslip, velocity and altitude histories for the simulated flight data. When NAVEQNS is propagated forward using the biased normal acceleration as input, the navigation equations diverge sharply from the measured data. This is marked as uncorrected data in Figs. 5 through 8.

After running LSIDNT, the corrected histories show excellent consistency with the simulated flight data. The corrected response is indistinguishable from the flight data in Figs. 5 through 8.

The second example demonstrates using IDEAS algorithms to update the longitudinal aerodynamic model of a twin turbofan military aircraft. The baseline simulation model is the operational flight trainer (OFT) code which is integrated with the IDEAS software. The modeling tool outlined above is used to add incremental non-linear angle of attack dependent terms to the basic pitching moment characteristics of the baseline model. In addition, simple bias corrections to elevator control power and pitch damping terms are identified.

The LSIDNT optimizer adjusts the specified parameters in the IDEAS model in conjunction with the baseline OFT simulation. Calibrated flight test data from a large number of test maneuvers are used to propagate the model and compare against model outputs. The final estimated model parameters are then used to compare the upgraded model response against the flight measured and baseline model responses. Figures 9 through 12 show this comparison for a low frequency longitudinal stick sinusoidal input maneuver. The measured elevator signal is used to excite the simulation model. These validation results indicate that although the upgraded model better represents the flight characteristics, additional model improvement is warranted.

Conclusions

The Integrated Data Evaluation and Analysis System provides a unique environment for complete flight test data analysis, simulation update, and validation. By gathering all of the tools needed for data preprocessing, system identification, and simulation into one processing environment, the flight test and simulation engineers can complete simulation updates more quickly. The system is designed to handle large quantities of flight test data and is setup for easy communication with several other analysis software packages.

Using off-the-shelf communication and data processing software provides for efficient development and operation. In addition, the API developed for IDEAS enable the users to take full advantage of its features when integrating new analysis tools with IDEAS.

The fundamental objective of the IDEAS program is to reduce the time and effort required to process flight test data, perform parameter identification, upgrade complex simulations, and validate the improved models. This reduction in time and the associated increase in productivity has been demonstrated in two recent and ongoing flight test data calibration and simulation model upgrade and evaluation tasks.

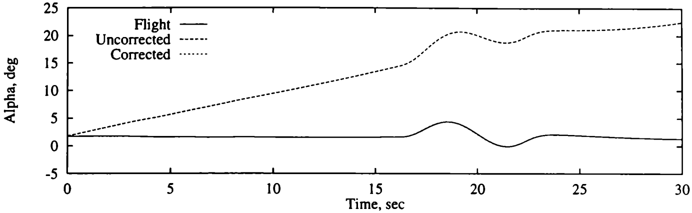


Figure 5. Comparison of Propagated Angle of Attack History
(Flight and corrected data coincide at this scale.)

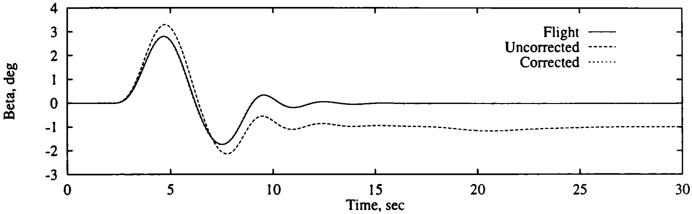


Figure 6. Comparison of Propagated Sideslip Angle History
(Flight and corrected data coincide at this scale.)

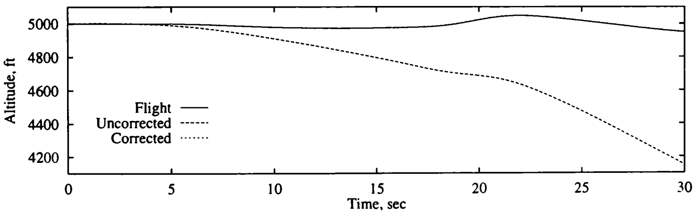


Figure 7. Comparison of Propagated Altitude History
(Flight and corrected data coincide at this scale.)

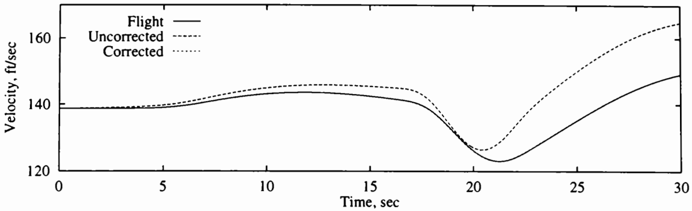


Figure 8. Comparison of Propagated Total Velocity History
(Flight and corrected data coincide at this scale.)

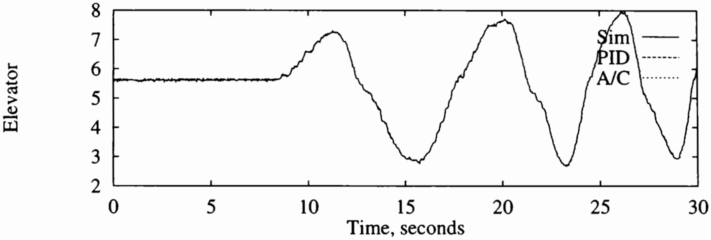


Figure 9 Flight Measured Elevator Angle

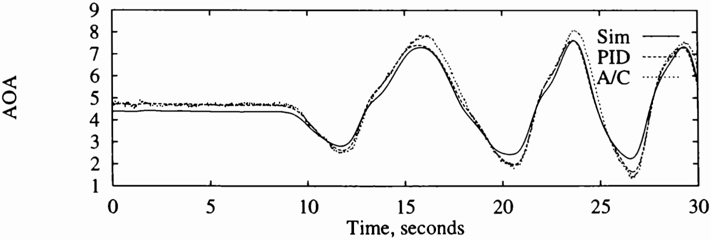


Figure 10 Angle of Attack Comparison
Flight data (A/C), Baseline simulation (Sim), Updated model (PID)

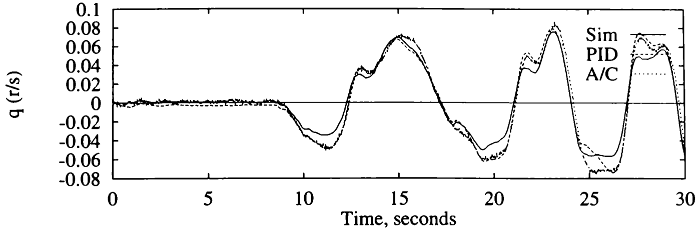


Figure 11 Pitch Rate Comparison
Measured (A/C), Baseline simulation (Sim), Updated model (PID)

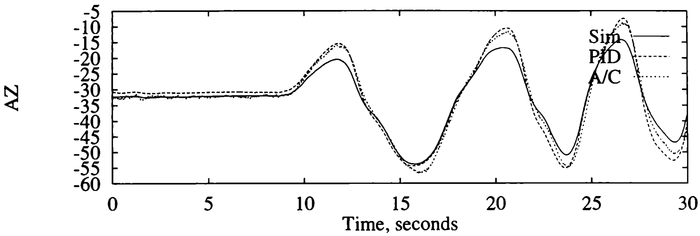


Figure 12 Normal Acceleration Comparison
Measured (A/C), Baseline simulation (Sim), Updated model (PID)

References

1. Maine, Richard E. and Iliff, Kenneth W., "User's Manual for MMLE3, A General FORTRAN Program for Maximum Likelihood Parameter Estimation," NASA TP-1563, 1980.
2. Maine, Richard E., "Programmer's Manual for MMLE3, A General FORTRAN Program for Maximum Likelihood Parameter Estimation," NASA TP-1690, 1981.
3. Bach, R. E., Jr., "State Estimation Applications in Aircraft Flight Data Analysis," *A User's Manual for SMACK*, NASA RP-1252, March 1991.
4. Tischler, Mark B., and Cauffman, Mavis G., "Comprehensive Identification from FrEquency Responses, Vol. 1 — Class Notes," NASA Conference Publication 10149, September, 1994.
5. Tischler, Mark B., and Cauffman, Mavis G., "Comprehensive Identification from FrEquency Responses, Vol. 2 — User's Manual," NASA Conference Publication 10150, September, 1994.
6. Anderson, Laurence C., "Robust Parameter Identification for Nonlinear Systems Using a Principal Components Regression Algorithm," AIAA-85-1766, AIAA 12th Atmospheric Flight Mechanics Conference, August 1985, pp 28–36.
7. Trankle, T.L., Rabin, U.H., and Vincent, J.H., "Filtering Flight Data Prior to Aerodynamic System Identification," AIAA Atmospheric Flight Mechanics Conference, August 1983.
8. Trankle, T.L., Vincent, J.H., and Frankin, S.N., "System Identification of Nonlinear Aerodynamic Models," Advances in the Techniques and Technology of the Application of Nonlinear Filters and Kalman Filters, AGARD-AG-256, Paper 7, March 1982.
9. Alaverdi, Omeed, Warner, Michael S., and Abanero, Jose N.T., "Development of a Flight Test Database for the MC-130E Aircrew Training Device," Proceedings of the 14th Interservice/Industry Training Systems and Education Conference, November 1992, pp. 197–204.
10. "NCSA Data Transfer Mechanism, Programming Manual," Version 2.3, National Center for Supercomputing Applications, University of Illinois at Urbana-Champaign, February 1992.
11. ANSI/ISO 9899-1990, *American National Standard for Programming Languages—C*
12. IEEE Std 1003.1-1990, *Information Technology—Portable Operating System Interface (POSIX)—Part 1: System Application Program Interface (API)*
13. Dennis, J.E. Jr., Gay, D.M., and Welsch, R.E., "An Adaptive Nonlinear Least-Squares Algorithm," *ACM Transactions on Mathematical Software*, Vol. 7, No. 3, September 1981, pp 348–368.
14. Giarrantano, J.C., "CLIPS v6.0 User's Guide," JSC-25013, NASA Johnson Space Center, Information Systems Directorate, Software Technology Branch, May 1993.
15. Fratter, C., and Stengel, R.F., "Identification of Aerodynamic Coefficients Using Flight Testing Data," AIAA-83-2099, AIAA Atmospheric Flight Mechanics Conference, August 1983.
16. Silhouette, X., and Stengel, R.F., "Estimation of the Aerodynamic Coefficients of the Navion Aircraft at High Angles of Attack and Sideslip," AIAA-87-2622, AIAA Atmospheric Flight Mechanics Conference, August 1987, pp. 452–463.

AIAA-96-3524-CP

SIMULATION SUPPORT OF A 17.5% SCALE F/A-18E/F REMOTELY PILOTED VEHICLE

Timothy R. Fitzgerald*
Naval Air Warfare Center Aircraft Division
Code 432200A, MS-3
48140 Standley Road
Patuxent River, Maryland 20670

David R. Gingras†
Science Applications International Corporation
Systems Technology Group
44417 Pecan Court, Suite B
California, Maryland 20619

Abstract

A simulation of a 17.5% scale F/A-18E/F remotely piloted vehicle (RPV) has been created and used for both engineering analysis and pilot training. The RPV team, comprised of personnel from Naval Air Warfare Center Aircraft Division, Patuxent River, North Carolina State University, and Bihrle Applied Research, Inc., has used the simulation as a tool for vehicle performance analyses, flying qualities assessments, and pilot training to reduce project risk. This paper details how the simulation was created and provides information pertaining to specific simulation models. The paper discusses the unique application of the simulation in real time using the Navy's Manned Flight Simulator facility. Lastly, the paper highlights results from a takeoff performance analysis which ultimately defined the takeoff configuration for the first flight of the RPV.

Introduction

In these times of shrinking budgets and the growing dependence upon simulation for flight research, aircraft development, and flight test support, the timely acquisition of aerodynamic data for use in high-fidelity simulation models has become increasingly important. Traditionally, data collected from wind-tunnel tests of sub-scale models have been the primary source of such simulation models, but these databases are assembled from a variety of test techniques, such as high- and low-speed static tests, rotary balance tests, and forced oscillation tests. Although necessary, the creation of an aerodynamics database with these data has several inherent draw backs. Dissimilarities among the data collection methods and inconsistencies in the tunnels themselves make creation of accurate simulation aerodynamics models challenging[1,2]. In addition, test data from wind-tunnels can be corrupted by wall and sting interference, flow variations, and blockage effects. A final drawback is that the scope of wind-tunnel testing can be limited because of the facility and test apparatus. This becomes a problem when attempting to model large variations in flow angle. For example, during F/A-18C/D departures, sideslip can exceed 45°[3] but the comprehensive collection of wind tunnel data at sideslips of this magnitude is difficult.

Technical issues are only part of the dilemma in using wind tunnel data for simulation model development; logistics can cause additional problems. With recent facility closures such as the 30- by 60-Foot Tunnel at the NASA Langley Research Center, wind tunnel entries may not be easy to obtain. Furthermore, the facilities that are available may not be as well-suited to the test requirements due to, for example, incompatibility of the test section with the model size. For these reasons, alternative, cost-effective methods for acquiring accurate aerodynamics data for the creation and improvement of simulation models must be explored.

One innovative solution for augmenting wind-tunnel data is through the use of data extracted from sub-scale, unpowered, free-flight models commonly known as drop models[4,5,6]. Although useful, these inertially-scaled free-flight models have historically been unable to generate large amounts of data in a timely fashion due to technical, logistic, and economic limitations. Recent advances in sub-scale propulsion, small-scale instrumentation systems[7] and software tools to aid parameter identification[8], have contributed to the feasibility of extracting aerodynamic data from such aircraft models in a timely manner.

A foreseeable advantage in using a powered sub-scale model to collect data for simulation validation is that, in addition to the greater flight duration when compared to a drop model, the rapid recovery and turn-around of a powered remotely piloted vehicle (RPV) will permit two to three approximately 15-minute data collection flights per flight day. By contrast, drop model programs have averaged one flight every 12 to 14 days, with a demonstrated capability to fly as frequently as once every three days. In addition, the average flight

* Aerospace Engineer
† Aerospace Engineer, Member AIAA

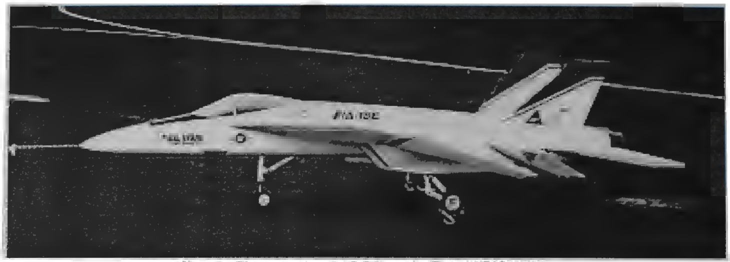


Fig. 1 Photograph of 17.5% scale F/A-18E/F RPV.

duration of a drop model is approximately 2 minutes[9]. In any given flight, a powered sub-scale model could potentially collect more than ten times the data than an unpowered drop model.

In addition to its ability to collect simulation validation data, a powered RPV may be well suited to compliment flight research using drop models. A powered RPV would enable test teams to investigate phenomena occurring in flight regimes that are typically difficult to reach with a drop model. For example, a powered RPV will be able to achieve prolonged straight and level flight as well as sustained accelerated turns, regimes where drop models are limited in capability.

The Naval Air Warfare Center Aircraft Division (NAWCAD), North Carolina State University (NCSU), Bihrle Applied Research, Inc. (BAR), and SWB Turbines, Inc. (SWB) have teamed to construct a 17.5% scale remotely piloted vehicle model of the F/A-18E/F (Fig. 1). The goal of the first phase of the project is to prove that a powered sub-scale model, equipped with the proper instrumentation system, can collect data useful for simulation aerodynamics model development and validation. Subsequent phases will focus on supporting simulation validation, F/A-18E/F flight test, and generalized flight research. An additional project goal is the acquisition of multiple wind tunnel entries with the RPV aeroshell to obtain a set of validation data. Using this controlled data set collected with the same aerodynamic model, flight data from the RPV may be directly correlated to an independent data source. This will permit a direct assessment of the feasibility of the RPV concept, and enable the quantification of unknown tunnel effects and/or induced engine and thrust effects. Unfortunately, as of this writing, the primary obstacle to this goal is the availability of a wind tunnel with a test section large enough to accommodate the RPV aeroshell.

Because of the relatively high risk associated with this research, the use of high-fidelity simulation was mandated. A six degree-of-freedom simulation of the RPV was derived from the full-scale F/A-18E/F simulation, interfaced with the Navy's Manned Flight Simulator (MFS) facility[10], and used for engineering analysis and pre-flight training purposes.

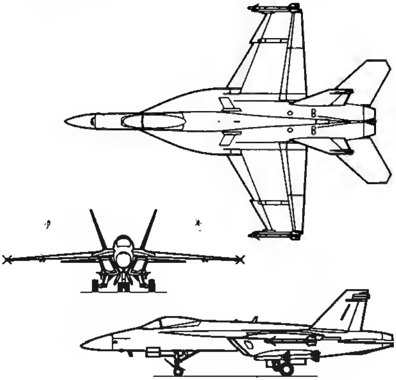


Fig. 2 Three-view diagram of F/A-18E/F.

F/A-18E/F 17.5% Scale Model

The F/A-18E/F RPV is a 130 lb, 17.5% geometrically-scaled model of the McDonnell Douglas F/A-18E/F Super Hornet (Fig. 2). The RPV was constructed using advanced composites by Foley Manufacturing, Inc., and is powered by two SWB-3 kerosene-burning turbojets that are each rated at 35 lbs static sea level thrust. State-of-the-art amateur radio control equipment is used to transmit pilot commands to ailerons, leading-edge (LEF) and trailing-edge flaps (TEF), rudders, stabilators, and engines; the leading-edge extension (LEX) vents and spoilers are currently inoperative and fixed in their undeflected positions. The RPV has fully retractable landing gear, complete with brakes and nose wheel steering. For the first series of test flights, an airspeed probe and engine RPM sensor will be used to measure a limited amount of information which will be recorded on board. Shortly after first flight, rate transducers, accelerometers, vertical gyros, angle-of-attack probes, pressure sensors, and telemetry will be added to the instrumentation system.

In later phases of the program, the RPV will be equipped with a video camera and various other systems to support ground-based cockpit operations.

RPV Simulation

The RPV simulation model was adapted from the full-scale, six degree-of-freedom F/A-18E/F simulation model developed under contract by McDonnell Douglas Aerospace and provided to the Navy. Both simulation models are capable of running in real-time, and use the Controls Analysis and Simulation Test Loop Environment (CASTLE) generic simulation architecture[11]. The simulation currently represents the RPV first-flight configuration, scheduled to be flown in late Spring of 1996.

Of the five main models used in the RPV simulation, the engine, weight and balance, and the controls system models were created by the Flight Vehicle Simulation Branch at NAWCAD using information provided by NCSU, BAR, and SWB. The landing gear and aerodynamics models used by the RPV simulation are slightly modified versions of the full-scale simulation models.

Aerodynamics Model

The RPV aerodynamics model was derived from the full-scale F/A-18E/F real-time simulation at NAWCAD. The only modifications made were to geometric constants such as mean aerodynamic chord, wing span, and reference area, and an adjustment to one of the independent variables in the ground effect function table which scaled this effect to the geometry of the RPV.

The F/A-18E/F aerodynamics model consists of a rigorous assortment of non-linear basic and incremental function table data in coefficient form. These data were collected from low- and high-speed static wind tunnel tests, with dynamic data provided by both forced oscillation and rotary balance tests. These data are linearly interpolated, and combined via a coefficient summary to produce the total force and moment coefficients acting on the airframe during each simulation time frame. Dynamic data are first combined using the Kalviste technique[12], then summed with the static data. Aerodynamic increments for a variety of store loadings, although not generally applicable to the RPV, are available.

Engine Model

The RPV engine model is also based on non-linear function tables constructed from static thrust and engine speed curves supplied by the RPV engine manufacturer, SWB. Dynamics are modeled via first-order lags. Radio transmission delays of the pilot inputs are also incorporated into the model. Malfunctions include single and dual engine failures.

Weight and Balance Model

The RPV weight and balance model is comprised of actual measurements of the RPV weight and center of gravity (c.g.), and estimates of the inertial characteristics. Initial approximations of pitch inertia have recently been validated to within 5% of measured data. Assuming that roll and yaw inertia estimates were of the same accuracy, a sensitivity analysis was conducted with the RPV simulation by varying the inertial characteristics in all three axes. This study revealed little discernible effect on predicted flying qualities, providing increased confidence in the use of the roll and yaw estimates for handling qualities work. These intertias will be measured prior to first flight. Provisions were made for full- and empty-weight conditions, with linear interpolation of the data between the two states to determine the weight and c.g. characteristics of partial-fuel conditions.

One typically minor aspect of full-scale weight and balance modeling that became important for the RPV simulation was the c.g. shift with main landing gear deflection. The main landing gear of the RPV is similar in geometry to the full-scale F/A-18E/F in that the main wheels pivot rearward as the landing gear strut is compressed. On the full-scale aircraft, the resulting c.g. shift is negligible, but in the case of the RPV, this equated to a c.g. shift of approximately 1.3% mean aerodynamic chord (MAC).

Control System Model

Figure 3 shows a block diagram representation of the RPV's first-flight control system as modeled in the simulation. For first flight, a simple control system was created that provided pitch control through collective stabilator deflection, roll control through differential aileron deflection, and yaw control through collective rudder deflection; all three axes are trimmable. Once again, radio transmission delays of pilot inputs are included. For the RPV, flaps were initially intended for take-off and landing use only (i.e., no scheduling of maneuvering flaps), so provisions were made in the simulation for half, full, and UP/AUTO (0° LEF/0° TEF) flap positions. As previously noted, the LEX vents and spoilers will not be functional on the RPV for the first series of flights, so these surface positions were "hard-wired" closed in the simulation. Simple failures of the primary control surfaces were modeled.

The RPV actuators are modeled in the simulation as second-order, rate-limited lags, and were based on time-response information of the JR4031, JR4421, and JR4721 electric servo motors selected for use in driving most of the RPV control surfaces. A model of the Condor PS-050 high power electric servo motor, used

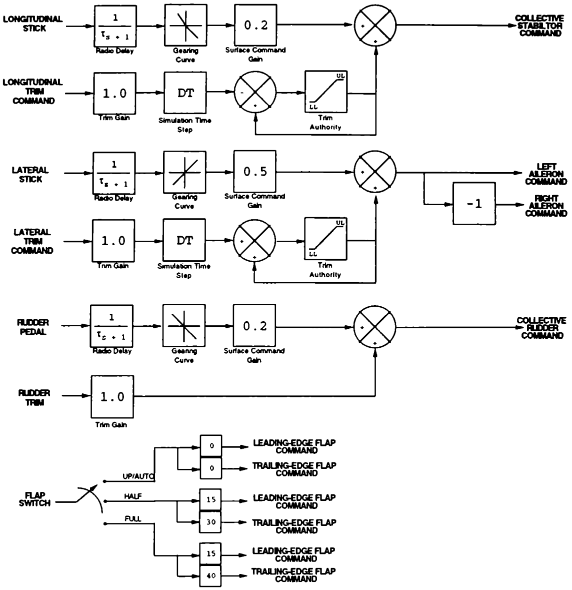


Fig. 3 Block diagram of RPV simulation control system model.

to drive the RPV flaps and stabilators, is currently under development by NCSU.

Landing Gear Model

The landing gear model used for the RPV simulation is a scaled-down version of the physically representative landing gear model used in the full-scale F/A-18E/F simulation at NAWCAD[13]. Each landing gear component, nose and main, is modeled as a single, decoupled strut-wheel assembly. RPV-specific strut reaction force and deflection data provided by NCSU were incorporated into the model. The results were then verified by NAWCAD using NCSU measurements.

Real-Time Interface

A unique aspect of the RPV simulation is the ability to interface it in real-time with an adapted radio controller (R/C) or one of the MFS facility's F/A-18 cockpits. The radio controller is similar to the one that will be used by the pilot during the initial phase of the RPV flight program. In this configuration, a fixed eyepoint location near one of the runways contained in the image generation (IG) database is presented, and the RPV simulation drives a moving model contained within the IG, indicating to the pilot the relative position of the RPV to that eyepoint. In the cockpit mode, the RPV simulation drives the image generation system in exactly the same way other MFS simulations do. This real-time interface will be used to support later phases of the program in which the RPV will be controlled by a pilot in a remote, ground-based cockpit.

Early in the RPV simulator development effort, the MFS 40-foot dome, with its target projectors and IG system, was used to provide visual cues to the pilot during real-time operations. The target projectors were slaved to the IG moving model, and were used in an effort to highlight the position of the RPV when it became too distant for the pilot to discern it from the rest of the IG visual scene. However, the ability of the target projectors to enhance the poor visual scene resolution was minimal, and their low reliability finally forced the abandonment of this set-up in favor of the facility's new helmet-mounted display (HMD) system. Using the HMD provides a more realistic environment, much greater visual scene resolution, and better system reliability, while only sacrificing some freedom of movement and the ability to quickly and easily glance down at the radio controller.

RPV Program Support

In addition to providing pilot training, the F/A-18E/F RPV simulation has been used to support the RPV first flight readiness program in several ways, primarily through the investigation of control power issues, the assessment of flying qualities, and the exploration of takeoff performance issues.

Engineering Analysis

Flying Qualities

Early use of the RPV simulation for engineering analysis quickly revealed potential flying qualities issues. The primary problem was static stability, both longitudinally and directionally. As would be expected when using the relaxed stability margins intended for the full-scale F/A-18E/F, the RPV became longitudinally unstable at very low angles-of-attack. Then, as the longitudinal instability caused angle-of-attack to rapidly increase, and with the flaps fixed in the 0°/0° UP/AUTO position, directional stability degraded to the point where the RPV would enter a violent, nose-slice departure. A secondary contributor to this was the over-abundance, primarily longitudinally, of control power, combined with small vehicle inertias. The stabilators provided enough pitch power to abruptly over-command RPV pitch response, which would result in undesired angle-of-attack excursions and airframe overstress. The solution to this problem was to reduce the control surface authority provided to the pilot while ensuring adequate static stability. In the case of the RPV, a c.g. position of 21% MAC provided acceptable longitudinal stability, and thus, departure resistance. After extensive simulation studies, gains of 0.2, 0.5, and 0.2 were placed on the longitudinal, lateral, and directional axes command paths respectively; these gains reduced the full-authority control surface commands modeled in the RPV simulation via the control system gearing curves. Since the original plan for the RPV did not call for scheduling of the leading- and trailing-edge flaps for first flight, initial flights will be restricted to half or full flaps. However, simulation studies indicate that by scheduling the RPV flaps with angle-of-attack, a marked increase in directional stability could be realized even at longitudinal static margins approaching zero. Such a flap schedule will be incorporated in the future.

Other analysis areas in which the simulation proved useful were in the extraction of linear aerodynamics models for control power assessments and Eigenvalue analysis of the dynamic stability modes of the RPV, the calculation of anticipated hinge moments to assist in accurate selection of the RPV actuators, and the determination of sensor bandwidth and resolution requirements.

Takeoff Performance Analysis

A study was conducted to determine the necessary amounts of thrust and pitch authority required to successfully operate the RPV from the NCSU flight facility. Initially, control surface combinations of 15° LEF, 20° TEF, no rudder toe-in, and ±5° of stabilator authority had been selected as the baseline takeoff configuration (15°/20°/0°). Table I contains the test matrix used in the investigation.

Table I. Real-time RPV minimum takeoff distance test matrix.

Configuration	Maximum Available Thrust (TMAX ≈ 68 lbs)	Maximum Stabilator Available
1	45% TMAX	±5.0°
2		±7.35°
3		±9.7°
4	55% TMAX	±5.0°
5		±7.35°
6		±9.7°
7	65% TMAX	±5.0°
8		±7.35°
9		±9.7°
10	75% TMAX	±5.0°
11		±7.35°
12		±9.7°
13	85% TMAX	±5.0°
14		±7.35°
15		±9.7°
16	95% TMAX	±5.0°
17		±7.35°
18		±9.7°
19	105% TMAX	±5.0°
20		±7.35°
21		±9.7°

Seven engine force levels, varying from 45% to 105% of the approximately 68 lbs total maximum rated static thrust (TMAX) were used, as well as three maximum stabilator deflections (±5.0°, ±7.35°, and ±9.7°), all of which employed an exponential gearing curve similar to the one available in the RPV radio controller. The real-time test procedure began with the RPV stationary on the runway with maximum power lever angle. Approximately 5 seconds into the ground roll, full aft stick was applied and held until rotation; the total elapsed time of the takeoff roll was noted. Once in the air, the pilot climbed steadily for several seconds, simulating the clearance of a 50-foot obstacle, then executed a 90° heading change. During the tests, the RPV pilot was required to make comments concerning the handling qualities for the 65%, 75%, 85%, and 95% TMAX cases, at all maximum stabilator deflections, based on his ability to control the RPV after takeoff, during climb out, and throughout the 90° heading change. Figure 4 contains a plot of minimum takeoff ground roll distances versus takeoff velocity for varying thrust levels and maximum stabilator deflections. It should be noted that the 45% TMAX line represents the minimum flyable thrust level. When this was combined with the ±9.7° maximum stabilator deflection, the RPV, although capable of rotating and climbing into the air, did not have enough thrust to power out of the resulting deep stall and maintain flight. In addition, it did not initially acquire enough altitude that could be "traded back" for airspeed before impact. Also of note regarding Fig. 4 is that, by using the engine manufacturer's thrust degradation curve as a function of ambient temperature, an equivalent %TMAX thrust level may be computed for any non-standard day. The resulting takeoff distance may be approximated through linear interpolation of the curves.

A second takeoff performance study consisted of performing pilot technique variations of the 65%, 75%, 85%, and 95% TMAX cases. For each of the 15°/20°/0° configurations, a gradual aft-stick pilot input (i.e., enough aft stick to rotate) was applied either 2 or 4 seconds after the recorded minimum takeoff times. This was done to study the handling qualities and takeoff distances with increased takeoff velocity using a more realistic pilot technique. These results are presented in Fig. 5. While handling qualities did improve as expected, there was a generally marked increase in the required takeoff distance.

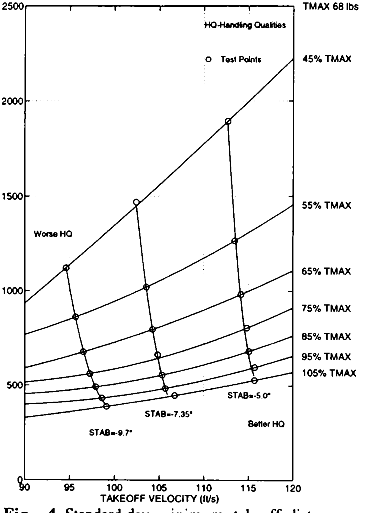


Fig. 4 Standard-day minimum takeoff distances for varying thrust levels and maximum stabilator deflections, (15°/20°/0°).

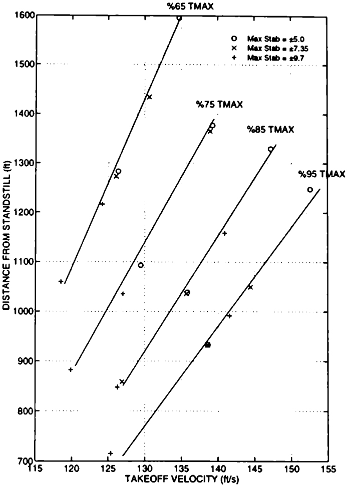


Fig. 5 Takeoff distances using delayed pilot input with varying thrust level and maximum available stabilator deflections, (15°/20°/0°).

A final takeoff performance study was conducted to explore aerodynamic methods for reducing the takeoff distance, which was becoming an increasing concern given the takeoff distances predicted for the baseline configuration during the previous two studies, and the 700-foot runway length of the intended base of operation. These included the use of varying amounts of rudder toe-in (10°, 20°, 30°, and 40°), or a 30° TEF deflection. Reductions in takeoff distance with each of these configuration changes were noted, as well as any pertinent handling qualities attributes. Based on these results, it was concluded that a configuration employing 15° LEF, 30° TEF, and 20° rudder toe-in (15°/30°/20°) would provide the best combination of takeoff performance and handling qualities. This configuration was further explored. Figure 6 contains a plot of minimum takeoff ground roll distances for this proposed configuration versus takeoff velocity for varying thrust levels and maximum stabilator deflections, which, like Fig. 4, may be interpolated for non-standard day conditions. In almost every case, the 15°/30°/20° configuration reduced takeoff distance while providing acceptable flying qualities, the single exception being the 45% TMAX case, in which the configuration

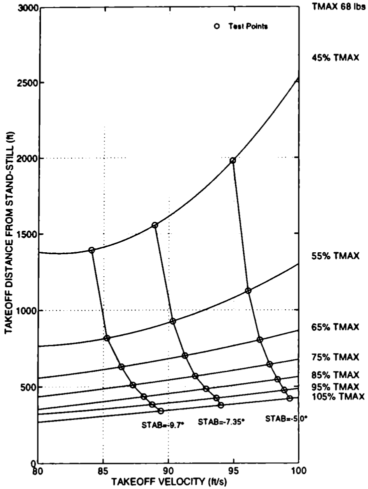


Fig. 6 Standard-day minimum takeoff distances for varying thrust levels and maximum stabilator deflections, (15°/30°/20°).

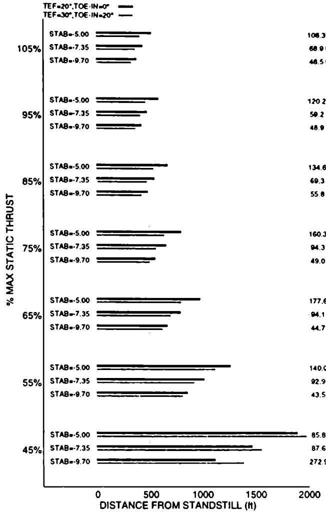


Fig. 7 Takeoff distance comparison between baseline and proposed configurations.

neither reduced takeoff distance nor was able to maintain flight. Figure 7 presents a direct comparison of minimum takeoff distance between the baseline configuration (15°/20°/0°) and the proposed configuration (15°/30°/20°).

Pilot Training

The F/A-18E/F RPV first flight test plan requires project pilots to fly the RPV simulation in real-time using the HMD interface at NAWCAD. During this training, the pilots were afforded the opportunity to familiarize themselves with predicted handling qualities of the first flight configuration, as well as develop and practice emergency procedures for a variety of failures.

Discussion of Results

The results of the takeoff investigation indicate that to achieve acceptable takeoff performance, the RPV's net engine force must be at least 75% TMAX. As Fig. 4 and 6 indicate, increased stabilator authority decreased the minimum ground roll, but for the 15°/20°/0° configuration, it also tended to degrade handling qualities. This is primarily the result of an over-abundance of pitch power for the given flap setting, which enabled the pilot to rotate the nose and take off before gaining sufficient airspeed for suitable flying qualities. However, this degradation in handling qualities was not apparent with the 15°/30°/20° configuration, making it desirable for reducing takeoff distance. Thus, it is recommended that the net engine force be at least 75% TMAX (although, net thrust forces in the 85% to 95% TMAX range are desired), takeoff flaps be set at 15° LEF and 30° TEF, rudders toed-in to 20°, and a maximum stabilator deflection of ±7.35° be employed (although as previously noted, ±9.7° would also be acceptable for this configuration).

Allowing the takeoff speeds to increase by delaying pilot input improved handling qualities at rotation and climb-out, but also increased ground roll significantly. Although a more realistic pilot input, this technique is not recommended for use with the 15°/20°/0° configuration due to runway length limitations at the intended base of operation. Use of this technique with the 15°/30°/20° configuration is more plausible, given its superior takeoff performance and generally acceptable handling qualities exhibited by all stabilator authorities.

Reaction of the project pilots to the simulator training was very positive. They found the resolution provided by the HMD adequate to the task of flying the RPV simulation despite the seeming bulk of the helmets and the lack of mobility that the system afforded. One problem that the pilots quickly noted was the poor contrast of the RPV visual model, which made it difficult for them to accurately discern its roll attitude. This was simply due to the default low-visibility color scheme of the visual system model, which can be corrected through the use of a color scheme that contrasts the ventral and dorsal sides of the visual model. With regards to the simulation model itself, although initially skeptical of the reduced control surface authorities that were proposed for the RPV, the pilots were pleased with the handling qualities and control harmonies exhibited by these gains. Control feel was good in all axes, turn coordination was excellent, and no departure tendencies were exhibited. As a direct result of this training, the pilots became comfortable that the simulation control gains were appropriate to use as the nominal RPV surface authorities for first flight. In failure scenarios, the pilots were given an invaluable opportunity to assess the affects of various control surface and propulsion system failures. Although it was impossible for them to directly discern the nature of each individual failure, this exercise gave them insight into how the RPV might globally react to failures and what difficulties they might experience in first determining that a failure existed, then in returning the RPV safely to the ground. In all cases, the pilots were able to quickly identify that some failure had occurred, and convince themselves that the RPV maintained sufficient controllability for a safe landing.

Summary and Conclusions

A simulation was created to model a powered 17.5% scale model of the F/A-18E/F. Engineering analyses using the simulation enabled the RPV team to rapidly evaluate the stability and controllability of the RPV while in flight, select hardware and sensors suitable to the anticipated flight envelope of the RPV, and enhance overall project safety through risk reduction. The RPV exhibits acceptable flying qualities in all configurations, and preliminary studies indicate handling qualities may be further improved, principally for high angle-of-attack flight, through the use of maneuvering (i.e., scheduled) flaps. Takeoff performance studies indicated that the RPV needs at least 75% TMAX, a trailing-edge flap deflection of 30°, and rudder toe-in of 20° to achieve adequate takeoff performance at the intended base of operation. The MFS real-time simulation promises to be a valuable asset for pilot training. The pilots have been able to develop and practice emergency procedures, as well as become familiar with the predicted handling qualities of the RPV.

Acknowledgments

The authors would like to acknowledge all participants in the F/A-18E/F RPV project, particularly the model builder, Foley Manufacturing Inc., and the engine manufacturer, SWB Turbines, Inc., as well as the NCSU graduate students who have provided timely information when needed throughout the development of this simulation.

References

[1] Fitzgerald, T. R., Ralston, J. N., and Hildreth, B. L., "Improvements to the Naval Air Warfare Center Aircraft Division's F/A-18 Subsonic Aerodynamic Model," AIAA 94-3400-CP, August 1994.

[2] Ralston, J. N. et al, "Evaluation of the NAWC/AD F/A-18C/D Simulation including Data Base Coverage and Dynamic Data Implementation Techniques," BAR 95-3, December 1995.

[3] Jaramillo, P. T. and Ralston, J. N., "Simulation and Analysis of the F/A-18D Falling Leaf Motion with an Assessment of Suppression Strategies," BAR 95-9, December 1995.

[4] Scher, S. H. and White, W. L., "Spin Tunnel Investigation of a 1/30-Scale Model of the McDonnell Douglas F/A-18 Airplane," NASA TM/SX-81809, August 1980.

[5] Fratello, D. J. et al, "Use of the Updated NASA Langley Radio-Controlled Drop-Model Technique for High-Alpha Studies of the X-29A Configuration," AIAA 87-2559-CP, August 1987.

[6] Croom, M. A. et al, "Dynamic Model Testing of the X-31 Configuration for High-Angle-of-Attack Flight Dynamics Research," AIAA 93-3674-CP, August 1993.

[7] Hall, C. E. Jr., "Instrumentation Array for a 17.5%-Scale F/A-18E RPV," (release pending) North Carolina State University, 1996.

[8] Linse, D. J., et al, "Integrated Software Tools for Simulation Update and Validation," AIAA 96-3523, August 1996.

[9] Croom, M. A., NASA Langley Research Center, personal communication, 22 April 1996.

[10] Burton, R. A., Miller, C. C., and Mills, R E., "Manned Flight Simulator and the Impact on Navy Weapons Systems Acquisition," AIAA-94-3420-CP, August 1994.

[11] Nichols, J., "The Generic Simulation Executive at Manned Flight Simulator," AIAA-94-3429-CP, August 1994.

[12] Kalviste, J., "Use of Rotary Balance and Forced Oscillation Test Data in a Six Degree-of-Freedom Simulation," AIAA 82-1364, August 1982.

[13] York, B. W. and Alaverdi, O., "A Physically Representative Aircraft Landing Gear Model for Real-Time Simulation," AIAA 96-3506, July 1996.

PRINCIPLES AND TECHNOLOGIES FOR REENGINEERING PILOT TRAINING

Joseph Sterling Mattoon
Personnel Research Psychologist

Armstrong Laboratory
Aircrew Training Research Division
6001 S. Power Rd., Bldg 558
Mesa, AZ 85206-0904

Abstract

Computer hardware and software technology has advanced far more rapidly than principles for guiding its application to aviation training. Technological capabilities will not increase the effectiveness or efficiency of training unless the structure and processes of current training programs are changed, and the specific potentialities of information technology are matched to student learning needs. Reengineering principles have been successful for increasing efficiency and productivity within business and industry and can be applied to the task of integrating new technologies within existing training programs. When combined with well-established findings of learning and training research, the reengineering of pilot training processes can successfully tap the power of new technologies to produce overall improvements in pilot training.

Computer hardware and software technology has advanced far more rapidly than principles for guiding its application to aviation training. For example, we are now able to simulate many types of flight environments with a great deal of accuracy and realism and record the actions, decisions, and behavior of pilots and aircrew in real time, but these new capabilities do not guarantee effective or efficient training. Part of the problem is due to older training models and strategies that were produced for a different type of economy with different constraints and limitations. Misconceptions and overgeneralizations of psychological principles of learning and educational research have also resulted in suboptimal training technology. Technologies, applied research, and reengineering principles should be combined to provide for a robust integration of technology rather than simply embellishing or automating parts of the existing training program.

Lessons from Education and Training Research

A recent study of the Air Force Air Education Training Command's Specialized Undergraduate Pilot Training (SUPT) revealed that few changes have been made in pilot training in 25 years (Andrews, et al., 1995). Results of this study indicate that improvements in SUPT with respect to cost and training time can be realized by implementing both off-the-shelf and specialized training hardware and software. In SUPT student pilots complete a fairly predictable sequence of activities that are driven by three major components--academic training (classroom lecture and computer-based training), flight simulator training, and flying training in the aircraft. The sequence seems simple to understand on a surface level--moving from academics to simulators to the aircraft, but the best approach to effective training involves a more complex and less predictive process. To optimize the value of training events during each undergraduate pilot training cycle, different training technologies should be applied in an iterative fashion that is driven by individual students needs rather than prescheduled facilities and activities. The value of any learning event depends on timing, what is learned during the event, and the effects of other events on the student's knowledge and skills. For example, an initial familiarization of the cockpit environment using the operational flight simulator (OFT) early in the training cycle may help students grasp the context and skills associated with detailed instrument training which occurs later in the cycle and is delivered in the classroom or by computer-based training (CBT). What may seem to be the most logical sequence of moving from simple to complex training delivery systems does not always hold true. The best training will incorporate many types of activities and a variety of training equipment in an iterative, and often times unplanned, fashion. Additionally, the optimal type, timing, and sequence of learning events may vary greatly across students. The implication is that optimal training must have flexible resources, include constant assessment of individual knowledge and performance, and adjust training activities to match each student's progress.

Methods for automated performance measurement and training management were suggested for Air Force pilot training over 25 years ago (Fowell, Rawlinson, Hirsch, & Hesse, 1971), but early computer technology was only capable of delivering simple instructional material, had very few visual display and interactive features, and offered no robust network capabilities. Computer technologies now provide high-resolution graphics, real-time simulations of complex systems, and the capability to communicate and exchange data from countless computer systems and users all over the world, but without an appropriate integration plan, even these powerful features do not guarantee effective or efficient training. Over 10 years ago, it became apparent to education and training researchers that computer- and visual media-based instruction holds no special magic that automatically improves learning (Clark, 1989). Computers and visual display systems can increase instructional efficiency-for acquiring certain types of knowledge, skills, and abilities, but this potential gain depends on the appropriate matching of instructional information and learning events to the immediate needs of each student. When technologies are applied to training haphazardly, as a function of their immediately appealing features or across the board using a "one-size-fits-all" approach, no significant gains in effectiveness or efficiency should be expected. Our current economy and associated training budgets have almost zero tolerance for shortsightedness in the procurement of expensive technologies, so jumping to acquire the latest gadgetry and then figuring out the best way to use it is no longer a viable strategy.

Training Research Plus Principles of Reengineering

Reengineering is a organizational strategy to increase effectiveness and efficiency by rethinking and reforming major processes that make up a recurrent effort to produce products or services (Hammer & Champy, 1993; Hammer & Stanton, 1995). Effective use of new technologies depends on the careful reengineering of pilot and aircrew training processes, because the original processes were designed for aviation systems, management structures, and economic factors that have changed dramatically in the last two decades. Thus, the improvement of training programs depends as much on reshaping old training processes as it does on implementing new and powerful enabling technologies.

Integrating Technology within the Training Process

To reap the full benefits of new technologies, substantial changes must be made in several aspects of training programs and the supporting organizations that maintain them. These changes are fundamental in nature and require strong support at all organizational levels. The following discussion will provide a theoretical justification for change and some recommendations on how training technology should be implemented rather than the logistics and political challenges that must be met to carry out training program reengineering.

The main principles of reengineering, which have been used to increase efficiency and productivity within business and industry, can also be applied to training programs. According to Hammer and Champy (1993), a major focus of reengineering is to leverage new information technology. They have proposed several important ideas, four of which I have modified for application to training programs. Following a short description of each reengineering principle, I discuss some instructional methods and technologies that could improve pilot training if they are properly integrated within the overall training process.

Identify the Entire Process The first step in reengineering is to identify and understand the entire process that is targeted for improvement. A process is defined as "... a collection of activities that takes one or more kinds of input and creates an output that is of value to the customer" (Hammer & Champy, 1993, p. 35). Unlike most approaches to organizational change, reengineering does not call for an exhaustive task analysis of the target process. Tasks represent the efforts that people currently engage in which may be ineffective, repetitive, or totally unnecessary to accomplish the desired outcome. Task analysis involves high costs and much time and effort but often results in a large quantity of relatively useless data and restricts the evaluator's thinking as a function of the very boundaries and limitations that may be eliminated by reengineering. The initial input to the Air Force pilot training process are untrained students, and the desired outcomes are pilots and aircrew who have the ability to successfully carry out Air Force missions. One might assume that the output is a specified number of graduates per training cycle, but this is only part of the overall process on the road to successful missions. For example, the initial capabilities and experiences of SUPT graduates will significantly affect the type and amount of additional training resources expended (e.g., flying hours) by the gaining units (Air Force units to which new pilots are assigned) to produce mission-ready pilots, so SUPT graduates represent one outcome of the overall training process that directly affects training expenditures needed for mission success. In this view, SUPT is a supporting process or component of the overall AF training process.

Beyond Automation When considering the application of new computer technologies, it is important to realize that automation alone is not the key to training efficiency. For example, CBT has often been viewed as a means to create electronic training manuals. This is not an effective use of the computer or the CBT training method. CBT has special capabilities that are not present within printed training materials, and its advantages are realized only when these capabilities are matched to actual human information processing needs and student learning behavior. Electronic "page turners" automate delivery of text and graphics but do not motivate the learner or facilitate acquisition of knowledge. In fact, written materials that were designed to be used in a hardcopy format lose informational and instructional utility when directly translated into an electronic format (Landauer, 1995). On the other hand, redesigning the most challenging learning material, so that it can be more easily understood via dynamic, visual demonstrations and mastered via interactive practice, applies the special capabilities of modern microcomputers in a manner that directly improves instructional effectiveness and efficiency (White, 1993).

Horizontal Compression One of the goals of reengineering is to reduce the number of people needed to complete a process or process component. The intent of horizontal compression is to increase efficiency by assigning each individual a more comprehensive role within a process. The immediate benefits are that personnel develop greater ownership, insight, and understanding of the process of which they are a part and are able to complete the process more effectively (higher quality outcomes) and efficiently (in less time) than can be accomplished when the process is fragmented among many people and departments. For example, tutors take on a more robust role than a lecturer in a student's learning process. The student and tutor develop a better understanding of what to expect from each other and how best to communicate and work together to reach the desired knowledge and skill objectives. Horizontal compression is the opposite of fragmentation. Consider pilot training programs that are compartmentalized (fragmented): one instructor teaches classroom course material; another monitors student performance in the operational flight simulator (OFT); and still another flies training sorties with students. The time/space gaps between these separate training components can restrict an instructor's ability to track student progress and obscure their perception of the linkage, interactions, and interdependency among different training activities. Compressing these training components may require that one instructor pilot (IP) accomplish a set of related training activities with the same small group of students. For example, the IP might conduct a prebrief on an instrument landing system (ILS) and associated procedures, coach students as they practice with desktop ILS simulations, then fly an ILS training sortie with each student. Compressing separate training components by integrating training activities would produce smooth progressions from initial fact and procedure learning to advanced skill development and would help IPs be more aware of what each student needs, identify when they need it, and be there to coach the student.

Vertical Compression Vertical compression refers to the practice of "empowering" people in a manner that gives them greater autonomy over major parts of the target process. In the context of pilot training, IPs would be provided with greater control over training resources such as changes/improvements in the design of training (e.g., user-centered design and formative evaluation), control over the use of flight simulators, and scheduling training aircraft. Few would argue that the eyes, ears, and professional judgments of IPs are the most important components of pilot training. IPs are not only instructors but also the role models, leaders, and performance evaluators of student pilots. Technology-based training may automate the delivery of some instruction to reduce the IP's load, but it should not replace or supersede the guidance of instructors or reduce the frequency of interpersonal communications among students and instructors. Reengineering should increase IPs' access and control of training technologies. Students also need to be given more responsibility for organizing and controlling their learning efforts as individuals and in small cooperative learning groups. This approach would help students build team skills and would reduce IPs' instructional delivery work load. Instructor time that is reclaimed from repetitive lecturing tasks can be applied to one-to-one tutoring and coaching.

Adaptive Training

As in most education and training programs, each SUPT student attends the same courses, receives the same content material, and therefore, spends about the same amount of time on each knowledge and skill objective. This "lock-step" method makes scheduling easier but produces less than optimal pilot training, because it fails to adequately address individual differences among students (Andrews, in press; Mattoon, in press). Research has shown that adaptive instruction such as one-to-one tutoring and interactive CBT can be more effective and efficient than group training for academic subjects (Bloom, 1984; Johansen & Tennyson, 1983; Ross & Rakow, 1981; Tennyson, 1981) and training complex skills (Fabiani, et al., 1989; Frederiksen & White, 1989; Mané, Adams, & Donchin, 1989). The rationale for designing adaptive instructional systems is based on the fact that student performance varies across different learning tasks, and students have different aptitudes for managing learning events (Snow, 1992).

There are many reasons for implementing adaptive training in pilot and aircrew training programs (Fishburne, et al., 1987; Mattoon, 1995). In adaptive training, the amount, type, and volume of instruction is adjusted to match each student's proficiency to enable the most capable students to advance quickly and to add instructional support to particular learning tasks in which an individual fails to make immediate progress. Group training schedules regulate advancement through each phase of training, so the best student pilots may be held back from achieving their highest potential, while less capable students may be pushed to more advanced training before they have actually mastered prerequisite skills. Training, aircraft maintenance, and other support contracts all function according to scheduled cycles, not because group training has proven to be the most effective or efficient system, but because it is an established and traditional public education and training model. The implementation of adaptive instruction and proficiency-based advancement would compensate for differences among students, but such a system would require major changes in both operating processes and management structure of SUPT.

To implement adaptive training on a large scale, each student's progress and performance would have to be closely tracked and stored in a database. One way to meet this requirement is to generate ongoing student proficiency profiles that identify specific training needs and describe student proficiency in each required knowledge and skill area. Profiles would be accessed by instructional devices (e.g., CBT and other automated training stations) to specify each new learning activity and by IPs to identify individual coaching or advisement needs. They would contain both hard data (e.g., test scores, response speed and accuracy on required tasks) and soft data (e.g., IPs ratings on training sorties and team member/leader effectiveness). A centralized relational database management system would distribute profile data to each training device or IP as needed during the training cycle. A student proficiency profile could become a permanent part of the pilot's record to provide information that is valuable for promotional consideration and career track assignment. For example, outstanding perceptual and motor skill capabilities may help identify the best candidates for fighter pilot tracks, while team management and interpersonal communication skills may have more value for working in larger aircrew environments. Thus, the student proficiency profile would be an essential component of adaptive SUPT training but would also serve a wider range of functions in the total training process.

Interactive Visual Displays

Pilots and aircrew are highly dependent on their ability to identify, interpret, and analyze visual cues (e.g., radar symbology), patterns (e.g., out-of-the-cockpit views of other aircraft and ground terrain), and configurations (e.g., multiship tactics), so repeated exposure to such visual information is necessary for both initial learning and skill maintenance. Interactive practice exercises that include exposure to appropriate visual information facilitates students' development mental models and pattern-recognition skills. Mental models, which are frequently based on imagery (Paivio & Linde, 1982; Rouse & Morris, 1986), are a major component of student pilots' understanding of the flight environment. A wide range of practice activities in which students interact with appropriate visual material (e.g., real-time simulation of cockpit instruments and out-of-the-cockpit views) are essential to mental model development. Cost, safety factors, and the graphic capabilities of microcomputer systems suggest that much of this kind of training should be completed on desktop instructional simulations (Mattoon, 1994).

Interactive CBT that features graphic-enhanced instruction and real-time simulation can help reduce "front loading" in pilot training. Front loading refers to the difficulty students have trying to acquire a large volume of new information prior to the opportunity to practice under operational conditions. When properly integrated with verbal instruction, visual diagrams, models, and simulations can significantly improve learners' acquisition of complex concepts and procedures (Mayer, 1989; Mayer & Sims, 1994; White, 1993), and the enhanced display capabilities of microcomputers can be used to leverage these instructional components. However, visual representation of all aviation concepts and tasks is hardly necessary nor cost effective. The findings of learning research and applied training principles should be used to identify the highest payoff areas--knowledge and skills that are best facilitated by interactive visual instruction. This approach will help differentiate training effectiveness and efficiency components from costly graphic embellishments that have little instructional value. The practice of transforming entire blocks of curriculum into CBT so that managers and administrators can claim they are "keeping current" and providing "state-of-the-art training technology" must end.

High-fidelity image generation systems and display technologies are now available for flight simulation at a much more affordable cost to training programs (Boyle & Edwards, 1992). Simulation displays can now be developed using affordable, commercially manufactured projection systems that provide partial or complete out-of-the-cockpit views (Thomas & Geltmacher, 1993). Also, head-mounted displays (HMDs) can completely immerse the student pilot within a three-dimensional, synthetic visual space (Thurman & Mattoon, 1994). Students can now use electronic training tools to observe and explore scaleable representations of systems and physical environments that were previously invisible (e.g., interior of an aircraft fuel system). Such displays, when combined with other hardware/software, enable students to visually rehearse an entire training mission or fly the mission in a portable simulator that can be wheeled in and out of a classroom almost as easily as a large piece of furniture. Such training devices no longer need special operators, technicians, or separate facilities that are normally regulated by restrictive schedules.

The proper implementation of training technologies goes far beyond the myopic goal of simply automating training. Training system developers must also consider portability, deployability, and adaptation to the changing needs of students and training programs. Information technologies can be leveraged to support both training management operations and special training configurations such as cooperative learning among student teams and guided coaching by IPs. For example, networked laptop trainers equipped with cellular modems could deliver instruction away from special facilities and support auxiliary functions such as immediate access to local procedures, communications among students and IPs, and just-in-time aircraft scheduling. The potential of such technologies are numerous, but the program infrastructure, organizational processes, and regulations that drive day-to-day operations must be shaped to exploit these capabilities.

Horizontal Compression of Pilot Training

Horizontally compressing pilot training programs can increase effectiveness and efficiency, but the training processes and operational infrastructure must be reshaped to tap the power of training technologies and the new kinds of training opportunities they afford. Many instrument-interpretation tasks, procedures, and mission planning tasks can now be practiced on laptop microcomputers (Nordwall, 1995). Larger systems that are equipped with special visual displays can provide practice on tasks that require out-of-the-cockpit views or simulations of full instrument panels. These trainers can be designed to be operated by the student without assistance from an instructor. Performance information can be collected, summarized, and transferred directly to student profiles or used (in real time) by the system to provide feedback, advice, and guidance on immediate performance or on choosing the most appropriate instructional events. To ensure that each student receives the right instruction at the right time, training devices need to pass information to and from the student proficiency profile database, and this requires a high degree of software compatibility and a high-performance network that links training devices and data management systems. Students should also be able to request assistance from IPs at any time, because many training interventions are beyond present automation capabilities. An effective coaching system, whereby IPs are continually on hand, requires some creative planning. The important goal is to make the most of IPs' time without forcing the process back into a group training mode. Group training is the best mode for certain parts of pilot training, but much more time should be devoted to providing opportunities for students to practice what they have learned and master prerequisite skills prior to advanced training in the aircraft. IPs' time must be balanced between groups of students and individuals who need coaching during practice on instrument procedures and aircraft control skills. A single learning center that is equipped with classroom, CBT, and simulation training hardware/software is probably the most viable configuration to accommodate variable training activities, adaptive instruction, and a wide range of knowledge/skill levels.

The most advanced stages of ground training involve practice and assessment of skills in a full-mission flight simulator. Many flight simulators were designed under the assumption that the closer the simulator matched the actual aircraft, the better it would train. This misconception can lead to suboptimal simulator training designs (Andrews, 1988). With the advancement of super minicomputers, image generators, and the capability to manage a full library of three-dimensional visual databases within the same training system, flight simulators can now accommodate the most advanced instructional interface functions and features. For example, it is now possible for computer voice-generation systems to deliver instructional guidance during practice, and voice-recognition systems to accept commands from students to initiate real-time changes during a training simulation scenario. For example, a student might command the training simulator to fly a model aerobatic maneuver while s/he simply observes the controls and instruments. When the student takes control of the simulator to engage in practice maneuvers, audio (voice) feedback can be activated to guide performance. This would minimize disruption of attention while providing ongoing feedback just as an IP would guide the student when performing in the aircraft. These "smart" training simulators can now be applied to basic flight training and will enable IPs to spend more time with students on the most complex problems and advanced flying strategies (Andrews, et al., 1995). The relative small size of minicomputers allow for portability throughout the training base, and network systems link simulators from different training sites to provide for distributed mission training--multi-ship and multiple-threat combat engagements that take place within the same simulated environment. When combined with in-flight mission recording systems, smart simulators would enable IPs to play back portions of an actual flight and have students re-fly particular maneuvers in the simulator that need additional practice (Andrews, et al., 1995). Smart simulators could draw information from a student's proficiency profile to suggest simulation scenarios that exhibit an appropriate level of challenge for the student's current skill level. The horizontal compression of advanced stages of training requires both the development of smart simulators that can be operated by the individual trainee and vertical compression of student and IP responsibilities.

Vertical Compression of Pilot Training

Vertical compression--greater empowerment and responsibility for IPs and students--arises as a natural outcome of horizontal compression. In a training program that has appropriately reengineered processes to integrate new technologies, IPs would no longer be restricted to particular components of the training cycle by separate courses, facilities, and schedules. Instead, they would support individual student needs as they occur and when and where they occur. This requires greater autonomy than IPs currently have in pilot training programs. IPs must be given the power to (1) choose how and where to use new training technologies, (2) move from one training activity and device to another as the need arises, and (3) advance individual students through training based on proficiency profile information and professional judgment rather than training cycle schedules. Students would also have greater responsibility and freedom within a reengineered training program. They would be responsible for managing their own study and training time, completing much prerequisite academic material on laptop systems, and meeting skill objectives that are monitored by automated training devices. These changes require new technologies plus full reengineering of the entire pilot training process.

Conclusions

New information technologies make it possible to increase training effectiveness and efficiency, but the full beneficial effect of technology cannot be realized by simply automating old tasks and portions of the training process. The entire training process must be reengineered in a manner that exploits technological capabilities by providing for horizontal and vertical compression and adaptive training that enables students to learn and build skills independently and advance from one phase of training to the next as a function of their individual proficiency. The greatest gains in training effectiveness and efficiency may be realized by the convergence of technology, reengineering, and training research.

References

Andrews, D. H. (1988). Relationships among simulators, training devices, and learning: A behavioral view. Educational Technology, 28(1), 48-54.

Andrews, D. H., Edwards, B. J., Mattoon, J. S., Thurman, R. A., Shinn, D., Carroll, L.

A., Moor, W. C., & Nelson, B. G. (1995). Potential modeling and simulation contributions to Air Education and Training Command flying training: Specialized Undergraduate Pilot Training (AL/HR-TR-1995-0157). Armstrong Laboratory, Aircrew Training Research Division, Mesa, AZ.

Andrews, D. H., Edwards, B. J., Mattoon, J. S., & Thurman, R. A. (in press). Potential modeling and simulation contributions to Specialized Undergraduate Pilot Training. Educational Technology.

Bloom, B. S. (1984). The 2 sigma problem: The search for methods of group instruction as effective as one-to-one tutoring. Educational Researcher, 13(1), 4-16.

Boyle, G. H., & Edwards, B. J. (1992). Low cost trainers: Lessons for the future. Proceedings of the 14th Interservice/Industry Training Systems and Education Conference (pp. 492-500). Orlando, FL.

Clark, R. E. (1989). Current progress and future directions for research in instructional technology. Educational Technology, Research, & Development, 37(1), 57-66.

Fabiani, M., Buckley, J., Gratton, G., Coles, M. G. H., Donchin, E., & Logie, R. (1989). The training of complex task performance. Acta Psychologica, 71(1-3), 259-299.

Fishburne, R. P., Jr., Williams, K. R., Chatt, J. A., & Spears, W. D. (1987). Design specification development for the C-130 model aircrew training system: Phase I report (Report No. AFHRL-TR-86-44). Operations Training Division, Williams Air Force Base, AZ.

Fowell, L. R., Rawlinson Jr., E., Hirsch, D. L., & Hesse, M. O. (1971). Future undergraduate pilot training system study (Northrop Operational Report NOR 70-149). Wright-Patterson Air Force Base, Ohio: Aeronautical Systems Division.

Frederiksen, J. R. & White, B. Y. (1989). An approach to training based on principled task decomposition. Acta Psychologica, 71(1-3), 89-146.

Hammer, M., & Champy, J. (1993). Reengineering the corporation. New York: HarperCollins.

Hammer, M., & Stanton, S. A. (1995). The reengineering revolution. New York: HarperCollins.

Johansen, K. J., & Tennyson, R. D. (1983). Effect of adaptive advisement on perception in learner-controlled, computer-based instruction using a rule-learning task. Educational Communications and Technology Journal, 31(4), 226-236.

Landauer, T. (1995). The trouble with computers. Cambridge, MA: MIT Press.

Mané, A. M., Adams, J. A., & Donchin, E. (1989). Adaptive and part-whole training in the acquisition of a complex perceptual-motor skill. Acta Psychologica, 71(1-3), 179-196.

Mattoon, J. S. (1994). Designing instructional simulations: Effects of instructional control and type of training task on developing display-interpretation skills. The International Journal of Aviation Psychology, 4(3), 189-209.

Mattoon, J. S. (1995). Reasons for implementing modeling and simulation technologies in Specialized Undergraduate Pilot Training (Report No. AL/HR-TR-1995-0078). AL/HRA, Mesa, AZ.

Mattoon, J. S. (in press). Modeling and simulation: A rationale for implementing new training technologies. Educational Technology.

Mayer, R. E. (1989). Models of understanding. Review of Educational Research, 59(1), 43-64.

Mayer, R. E., & Sims, V. K. (1994). For whom is a picture worth a thousand words? Extensions of a dual-coding theory of multimedia learning. Journal of Educational Psychology, 86(3), 389-401.

Nordwall, B. D. (1995, July 3). Navy to use laptops for pilot training. Aviation Week & Space Technology, pp. 68-69.

Paivio, A., & Linde, J. (1982). Imagery, memory, and the brain. Canadian Journal of Psychology, 36(2), 243-272.

Ross, S. M., & Rakow, E. A. (1981). Learner control versus program control as adaptive strategies for selection of instructional support on math rules. Journal of Educational Psychology, 73(5), 745-753.

Rouse, W. B., & Morris, N. M. (1986). On looking into the black box: Prospects and limits in the search for mental models. Psychological Bulletin, 100(3), 349-363.

Snow, R. (1992). Aptitude theory: Yesterday, today, and tomorrow. Educational Psychologist, 27(1), 5-32.

Tennyson, R. D. (1981). Use of adaptive information for advisement in learning concepts and rules using computer-assisted instruction. American Educational Research Journal, 18(4), 425-438.

Thomas, M., & Geltmacher, H. (1993). Combat simulator display development. Information display, 9(4 & 5), 23-26.

Thurman, R. A., & Mattoon, J. S. (1991, December). Designing microcomputer-based instructional simulations: Implications from cognitive psychology. Paper presented at the 33rd International Conference of the Association for the Development of Computer-Based Instructional Systems, St. Louis, MO.

Thurman, R. A., & Mattoon, J. S. (1994). Virtual reality: Toward fundamental improvements in simulation-based training. Educational Technology, 34(8), 56-64.

Thomas, M., & Geltmacher, H. (1993). Combat simulator display development. Information display, 9(4 & 5), 23-26.

White, B. (1993). ThinkerTools: Causal models, conceptual change, and science education. Cognition and Instruction, 10(1), 1-100.

AIAA-96-3527-CP

USING AN INDEPENDENT DATA ACQUISITION SYSTEM TO EVALUATE A SIMULATOR'S CUEING SYSTEMS

William M. Waldron*
Science Applications International Corporation
California, Maryland

Abstract

With increasing simulation capabilities and requirements the importance of simulator testing and evaluation is constantly increasing. In order to ensure proper cues are being presented to the pilot a thorough knowledge of the dynamics of the cueing systems must be acquired. To assist in acquiring this knowledge and to determine if the achieved and commanded cues are consistent, a method of instrumenting the simulator is required. This method must provide highly accurate and repeatable measurements.

This paper will discuss the measurement of end-to-end and subsystem delays and time and frequency responses of cueing systems using the PC based Simulator Evaluation System (SIMES). This system uses preprogrammed test procedures, independent sensors, and deterministic acquisition under a real-time operating system to provide a consistent and repeatable means of evaluating the performance of the simulator cueing systems. The cueing systems tested include the visual, motion, control loading, and cockpit instrument systems. In addition, an interface with the host computer memory is established to provide access to internal parameters in the host. This interface allows both reading and overdriving of host parameters. Results are presented which were obtained during a quantitative evaluation of a simulator.

Introduction

The cost of training simulators can easily run into the millions of dollars, however this cost can be justified if the simulator performance can be established. For example, a commercial training simulator must meet or exceed all cueing requirements for training as determined by the FAA. Such tests are usually complicated and time consuming. It is important to be able to quantitatively test the performance of a simulator and its cueing systems.

The Simulator Evaluation System (SIMES) is designed specifically to instrument and test simulators using instrumentation that is separate from those integrated into the simulator device for its operation. This paper will discuss the general architecture of SIMES, discuss the tests performed on a simulator by SIMES, and present some test results from a evaluation performed by SIMES.

SIMES System

The overall idea of SIMES is to provide an easy, yet comprehensive method of independently testing simulator cueing systems. SIMES is designed to accomplish this task through the use of a graphical user interface to run the tests, off-the-self hardware and software to perform the tests, and automatic logging and test documentation to record the output of the tests. SIMES is also designed to be portable, so it can be moved from site to site with the test team.

SIMES is comprised of a personal computer (PC) with an Intel "Pentium" processor, a reflected memory network to interface with the simulator host computer, and its own sensors and I/O capability. SIMES is built on the PC platform due to its forward compatibility, ease of use, and the abundance of software and hardware designed for this type of system. In addition, since PC systems are so widely used, future expansion and adaptation to include other types of testing and other system sensors will be straightforward.

The design objective of the SIMES system was to develop a system independent of the simulator being tested, capable of highly accurate measurements, able to access the host computer memory, easy to use, and comprised of commercially available components. The objective of being independent of the tested simulator is important for two reasons. This independence will allow validation of the simulators built in sensors and will allow SIMES to be moved between many different simulators. The high accuracy of the sensors, and in

[*] Senior Member, AIAA

fact the data acquisition system as a whole, is required to obtain the detailed measurements necessary to meet validation or acceptance testing specifications on the simulator. Access to the host memory is important in order to read and overdrive variables in the host in order to thoroughly test the different cueing systems. More on the access to the host memory will be discussed later in this paper. The SIMES system must also be easy to use. This ease of use is attained in three ways:

1) Consistent man/machine interface
2) Emphasis of graphical output
3) Computer prompted operation

Commercially available components of the SIMES system are used in order to keep costs down and to take advantage of technologies already developed. As a result the development of SIMES involved extensive integration of many commercial components. Since the SIMES architecture was designed to be flexible enough to accept the integration of many different components the capacity for future expansion was built into the SIMES system. The general architecture of the SIMES system is shown in Figure 1.

Key Components

The SIMES system is comprised of two key components; the data acquisition system and the graphical user interface. The data acquisition system records measurements from the instrumentation and provides the means to input to the host computer through a reflected memory network. The user interface provides the displays and prompts the user through preprogrammed test procedures to evaluate a simulator.

Data Acquisition

The data acquisition system runs under the real-time operating system iRMX® developed by the Intel Corporation. The data acquisition system running deterministically under iRMX® acquires data from typical analog sensors such as force and position sensors and accelerometers. In addition, the data acquisition system is configured to read and overdrive the variables in the simulators host memory through the Host Data Interface (HDI) and acquires information about the displayed visual scene using a specialized sensor called the Electronic Visual Display Attitude Sensor (EVDAS). Both the HDI and EVDAS systems will be discussed later in this paper. The analog inputs are differentially acquired using two 16 bit A/D boards at a nominal acquisition rate of 1000 Hz per channel. This fast acquisition rate is more than adequate to capture the dynamics of the simulator cueing systems and allows a one millisecond resolution in time delay measurements. The data acquisition system also time stamps each piece of data acquired.

The HDI provides the means for SIMES to acquire and/or overdrive the simulator's internal variables. This capability of the HDI allows the different systems of the

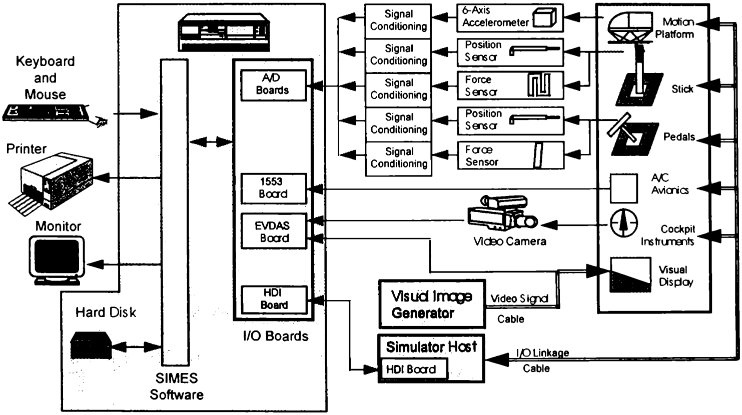


Figure 1: General Architecture of the SIMES System

simulator to be tested independently. For example, by instrumenting the control stick with a position sensor and using the HDI to monitor the variable which represents stick position, the time delay between a movement of the stick and the host computer getting this movement information can be calculated. In addition, with the overdrive capability of the HDI, the output cueing systems can be tested independent of the rest of the simulator's systems.

The sensor used to instrument the visual system is the Electronic Visual Display Attitude Sensor (EVDAS).[1] This system was developed by Wright Laboratories, Wright Patterson AFB and analyses, using analog hardware, the video signal from the simulator image generator. EVDAS provides information from which the pitch and roll of the visual scene can be calculated. EVDAS accepts any RGB video signal and outputs the X and Y location of the edge of a selected color within two user defined "sensor stripes". For example, if the two sensor stripes are configured to run vertically along the left and right edges of the display and are calibrated to monitor the sky color, EVDAS will provide the X and Y coordinates of the last occurrence of the sky color within each stripe. This information is then used to calculate the pitch and roll of the visual scene. Figure 2 shows the sensor strips superimposed on the visual scene.

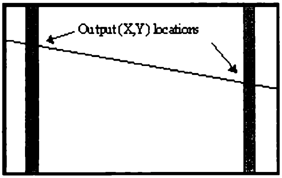


Figure 2: EVDAS Sensor Stripes

Graphical User Interface

The graphical user interface provides preprogrammed test procedures to test the simulator. This interface runs under Microsoft® Windows™ 3.1 which runs as a task under the iRMX® real-time operating system. To the user the system appears to be a standard Windows™ installation. The graphical interface presented to the user while performing the simulation tests was developed using a software package called TestPoint by Capital Equipment Corporation. This interface will prompt the user through the setup, configuration, calibration, acquisition, analysis, and report generation steps of the preprogrammed tests. During the procedure steps, help screens are presented to the user to assist in completing the test. During the report generation step of the test procedure, the TestPoint application will automatically generate a report using Microsoft® Word and Excel. This report is comprised of data stored during the previous procedure steps including graphs of the acquired data. This report serves as a written record of the test performed that can be incorporated into an overall test document or left as a stand-alone report of a particular test. Using Windows™ for the interface allows access to an extremely large base of software that can be used to provide additional analysis of the data that is not part of the preprogrammed test. In addition, the use of Windows™ allows easy access to the data which is stored in ASCII text files.

Simulator Tests

SIMES was designed to test the different primary cueing systems as well as the math model of the simulator. The cueing systems which can be tested with the baseline SIMES system include the visual, motion, control loading, and flight instrument systems. These systems can be tested individually or as part of the larger simulator in end-to-end tests. Additional cueing systems can be tested by attaching sensors appropriate for that cueing system to the SIMES data acquisition system.

Visual Tests

The visual cueing system of a simulator provides to the pilot a spatial reference between the aircraft and its environment under most conditions of simulated flight. For this reason, the visual system is one of the most important cueing systems in the simulator and hence good visual system performance can have a significant effect on simulating the flight of an aircraft. There are two tests that will be performed on the visual cueing system by the baseline SIMES system. These tests determine throughput delay and update rate of the visual cueing system.

The throughput delay is defined as the time it takes from the point at which the eyepoint data is available at the image generator interface with the simulator host computer to the end of the visual frame when this information is presented by the display system [2]. The eyepoint data is input to the simulator using the host data interface and the display response is measured by the EVDAS system.

The visual subsystem update rate test determines if the frequency at which the display is updated with new state vector information is constant. The test is accomplished by changing the eyepoint data so that new eyepoint information is available to the image generator at every cycle of the host. This new information will be processed by the visual subsystem and used to update the displayed scene. The changes in the displayed scene will be measured by the EVDAS unit along with the time between these changes. The intent of this test is to determine if the image generator uses new information for each visual frame displayed or if multiple visual frames are displayed using the same eyepoint data.

Motion Tests

In order to test the motion system of a simulator, linear and angular accelerometer packages are mounted onto the motion platform at known locations. Each package is comprised of three accelerometers, either linear or angular, aligned in mutually orthogonal orientations. The six accelerations are acquired by the SIMES data acquisition system. In addition, the leg positions are acquired by tapping into the measurements available within the motion control box. Independent sensors were not used because the internal measurements of the leg positions are usually much more accurate than what could be obtained with an external sensor. The acquired platform accelerations and positions are transformed to the reference location and coordinate frame for analysis.

The tests of the motion system fall into two categories; time domain and frequency domain tests. The time domain tests are comprised of accuracy, smoothness, crosscoupling, and time response tests. The frequency domain tests consist of frequency response and vibration power spectrum tests. The motion system tests are conducted on one axis at a time and can be performed either with or without the motion cueing algorithms active.

Control Loading Tests

Significant effort is involved in ensuring that the simulator cockpit controls provide the same "feel" as those in the actual aircraft. To determine if this effort has been successful, tests are performed on the cockpit controls. Two such tests are the Force vs. Displacement and the Pull and Release tests. These tests are performed as outlined in FAA Advisory Circular 120-40C.

The force vs. displacement test is performed in both degrees of freedom of the control stick (or column and wheel degrees of freedom for transport aircraft) and on the rudder pedals. Each degree of freedom of the controls is tested independently. The force vs. displacement plot can be compared with reference data such as data taken from an actual aircraft if such data is available.

The pull and release test is intended to test the time domain dynamic response of the control loading system. The stick is displaced to a position between 25% to 50% of the full range of motion. After holding this position momentarily the control stick is released and the dynamic response is recorded. This response can be compared with reference data.

Flight Instrument Tests

The flight instrument tests are intended for artificial horizon and "steam gage" type instruments. These tests are performed in a manner similar to the visual system tests. If the instruments are "glass cockpit" displays and the video signal is available, this signal can be input into EVDAS and throughput delay and update rate tests can be performed identical to the visual system tests. If the instrument does not have a video signal available a video camera can be used to monitor the instrument and the camera video signal is used to input into EVDAS to perform the tests. In addition, if the instrument is driven by a simple analog signal this signal can be acquired by one of the A/D boards to perform the tests on the flight instruments.

End-to-End Tests

The end-to-end tests are designed to evaluate the transport delays within the simulator system and the frequency response of the system. A step or sine wave is input to the control loading system and this signal is allowed to propagate throughout the simulator systems. If the aircraft accelerations are set proportional to the control loading input, the simulated dynamics of the aircraft can be removed from the resulting measurement. The resulting transport delay or frequency response will then represent the simulator hardware and architecture and not include the modeled response of the aircraft. By using the HDI, data at intermediate points within the system is obtained so that not only end-to-end results are obtained but also delays and the frequency responses between different system components.

Math Model Tests

Since the HDI gives access to the internal variables within the host computer, the simulation math model can be instrumented. This access gives large flexibility to the types of tests that can be performed to obtain data on the math model of the vehicle and its modeled systems. The HDI also gives the ability to overdrive the simulation with recorded data such as obtained during a flight test in order to validate the math model. This overdrive capability can be a very valuable tool if the simulation architecture is not configured to perform this overdrive function itself.

Test Results

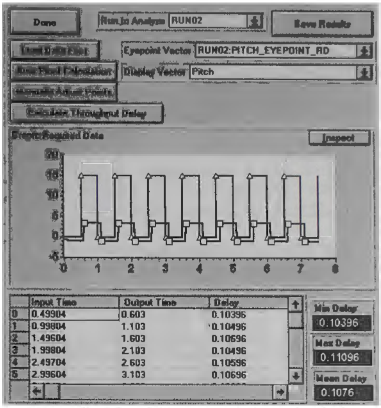
The test results presented below were obtained on a modern high fidelity military simulator. The results from two of the tests performed during this evaluation, a visual throughput delay and a control loading pull and release test, will be presented.

Visual Throughput Delay

The visual throughput delay test was performed by using the HDI to overdrive the pitch eyepoint variable with a function that toggles between zero and fifteen degrees at one-half second intervals. This input is accomplished by the SIMES data acquisition system by toggling an intermediate variable on the reflected memory network. The simulation code had been modified such that when SIMES was operating in overdrive mode the host variable for pitch eyepoint would be set to the SIMES intermediate variable at the time that the host pitch eyepoint variable is normally made available to the image generator. Immediately after this overdrive of the host pitch eyepoint variable, a second intermediate variable is set equal to the newly overdriven host pitch eyepoint and echoed back out to the shared memory network. This second intermediate variable is used to determine when the host variable for the pitch eyepoint was overdriven and hence when a new value was available to the image generator.

EVDAS was used to monitor the visual signal going from the image generator to the image projectors. By analyzing the visual signal using analog hardware, EVDAS can be used to determine the horizon position in the visual scene. EVDAS outputs the visual scene data at the end of every video frame. By acquiring the echoed value of the pitch eyepoint on the HDI and the horizon position as determined from the EVDAS data the throughput delay can be calculated. The time for the projectors to display the visual scene is neglected.

Shown in Figure 3 is the graphical interface analysis window for the visual throughput delay preprogrammed test. The toggling of the pitch eyepoint and displayed pitch can be seen in the graph of the acquired data. A measurement of the throughput delay is obtained at each toggle point. For this example, the visual throughput delay ranged from approximately 104 to 111 milliseconds with a mean of approximately 108 milliseconds.



Run to Analyze: RUN02

Eyepoint Vector: RUN02:PITCH_EYEPOINT_RD

Display Vector: Pitch

	Input Time	Output Time	Delay
0	0.49904	0.603	0.10396
1	0.99804	1.103	0.10496
2	1.49604	1.603	0.10696
3	1.99804	2.103	0.10496
4	2.49704	2.603	0.10596
5	2.99604	3.103	0.10696

Min Delay: 0.10396

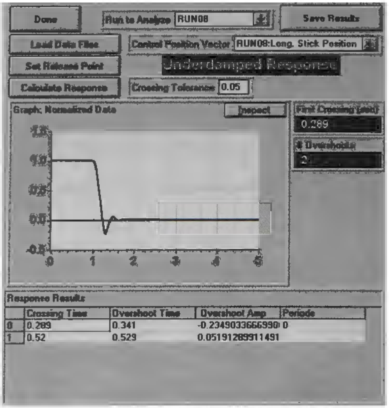
Max Delay: 0.11096

Mean Delay: 0.1076

Figure 3: Visual Throughput Delay Analysis Window

Control Loading Pull and Release

The control loading pull and release test was performed as outlined in FAA Advisory Circular 120-40C. To perform this test the control stick was instrumented with a high accuracy position sensor call FASTAR. FASTAR is a precision variable induction transducer manufactured by Data Instruments and is used like an LVDT. For this example, the control stick was displaced approximately 50% of full motion in the longitudinal axis only, momentarily held in this position, and the released. The position was acquired at a rate of 1000 Hz. During the analysis, the response was identified as underdamped and normalized for easier comparison with reference or other test data. Since the response was underdamped, crossing times, overshoot times and amplitudes, and response period are calculated as described in FAA Advisory Circular 120-40C using a user specified crossing tolerance of ± 5%. Figure 4 shows an example analysis window for the control loading pull and release test. If the response would have been overdamped, the time to 90% decay would have been calculated and displayed.



Run to Analyze: RUN08

Control Position Vector: RUN08:Long. Stick Position

Underdamped Response

Crossing Tolerance: 0.05

First Crossing (sec): 0.289

Overshoots: 2

Response Results

	Crossing Time	Overshoot Time	Overshoot Amp	Periode
0	0.289	0.341	-0.2349033666990	0
1	0.52	0.529	0.05191289911491	

Figure 4: Control Loading Pull and Release Analysis Window

Conclusion

As discussed, using an independent data acquisition system provides the capability to thoroughly test the simulators cueing systems as well as provide access to the host computer internal variables. Using an independent system allows acquisition rates to be run much faster than the host cycle time thus allowing more detailed information to be gained on delays in the system. Another advantage of having an independent data acquisition system is that one system can be used to test many simulators, thus offering a tremendous cost savings over incorporating the test capabilities into each simulator.

References

1. Slutz, G. J. and Ewart, R. B., "An Electronic Visual Display Attitude Sensor (EVDAS) for Analysis of Flight Simulator Delays", AIAA-92-4167-CP.

2. FAA Advisory Circular 120-40C (DRAFT), July 1995.

3. Waldron, W. M. and Schwalbe G. C., "Simulator Evaluation Using A Portable Integrated Package of Independent Sensors", AIAA-95-3400-CP.

AIAA-96-3528-CP

SIMULATOR CUE VALIDATION USING FREQUENCY RESPONSE TECHNIQUES

R. Thomas Galloway + and R. Brad Smith++
Naval Air Warfare Center Training Systems Division
Orlando, Florida

Abstract

This paper documents the application of the piloted frequency sweep technique, and subsequent frequency response analysis, to quantitatively identify the cue-sync character of USMC rotary wing training systems. Frequency response comparisons were generated by analysis of data generated using the piloted frequency sweep technique. The frequency response comparisons were used to evaluate the simulator motion and visual systems versus the simulator model thereby providing a means for identification of cue-sync character without the need for special purpose software modifications. Simulator frequency response parameters were produced that are equivalent to evolving aircraft handling qualities criteria from ADS-33C and MIL-STD-1797A which will facilitate simulator validation with respect to aircraft flight characteristics. Finally, a quantitative evaluation criteria for motion system cueing quality is provided in order to assist in the determination of motion system character.

Introduction

Simulated flight training system users and vendors in the simulation industry understand well the importance of proper presentation of major sensory cues to a trainee within a simulated flight training system. Motion, visual, crew compartment displays and instrumentation form the primary cues delivered to a trainee in modern day flight training systems. Crucial to proper presentation of cues is having a quantitative means to measure the character of the cue. Key to proper characterization of the cues is making the measurement in such a way that relationship, or synchronization with the other the relationship, or synchronization, with the other primary cues can be quantitatively determined. Measurement of cue-synchronization (cue-sync) character is the topic of this paper.

The necessity to measure cue-sync arises as a result of the knowledge that all cues may not arrive at the trainee station simultaneously. Delays within the various subsystems of the training system architecture are the culprit. Some delays are necessary and/or desirable, others are not. Delays result from a number of sources: data transfer time amongst the various subsystems of the architecture, processing time, sampling delays, model response, and hardware response. Data transfer time is the time required to move information from one subsystem to another. Typically, while information is being transferred, it cannot be utilized. Processing time is the time required for the computational system to produce an update to the solution of aircraft state. Sampling delays result because in a digital simulation, processing time occurs in discrete time intervals or frames. Inputs to the simulation are delivered in one frame, processing occurs in subsequent frames, and outputs are delivered to the cueing systems in succeeding frames. Sampling delays can be affected by data transfer delays if the transfers are asynchronous in nature. Utilization of synchronous transfer methods reduce this effect. Transfer delays can be grouped with processing time if the transfer can take place synchronously and can be accomplished in less

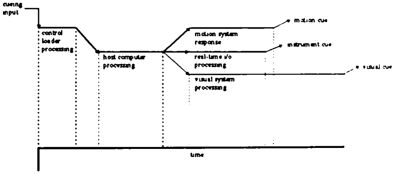


Figure 1 - Cue-Sync Relationships

+ Aerospace Engineer, Member AIAA
++Electronics Engineer

than the time of one frame. Figure 1 provides a simple illustration of the above concepts. The proportions indicated are not to scale.

Current Methods

Some methods [1] to determine the cue-sync character of simulator primary cueing systems involve the use of special purpose test software which bypass the aerodynamic model delays in the simulation and other cue compensation effects. The special purpose test software is placed downstream of the flight and cue dynamic models and acts to overwrite the normal results that determine and update the aircraft state so that their contribution to processing time will remain a component of the throughput measurement without the delays due to the dynamic properties of the models and any cue compensators present. The various control axes are driven (at a position downstream of the actual control in some implementations which neglects a portion of the system response) with a step input while measuring the time to first response of the primary output cues. The system is considered adequate if the response time measured falls below a certain delta time with respect to each other. A variation on the technique also provides for the application of *fixed* frequency input sinusoids to the various control axes while making the same or similar output measurements in an attempt to identify any gain and/or phase distortion as a function of frequency. Subsequent comparison of the output measurements determine whether the system is considered adequate.

One negative aspect of the above technique is the necessity to have special purpose software in place in order to make the desired cue-sync measurement. This can be especially disappointing if the design of the software architecture of the training system did not envision a requirement to perform special purpose testing of this nature. Another negative aspect of the above technique is that with the aerodynamics model "short-circuited" in this manner, a significant portion of the training simulation is missing. In other words, it is tested in a configuration that is different from that which is normal for training purposes.

A New Approach

A method to measure the cue-sync character in a manner that does not disturb the training system in its training configuration is desired. The method should allow for complete characterization of the training system total end-to-end response in any or all of the cueing axes. Additionally, the cue-sync measurement method should be compatible with normal aircraft flight test techniques.

It is the assertion of the authors that determination of cue-sync character of a simulated flight training system can be obtained *exclusively* through use of the piloted frequency sweep technique. The frequency response, produced from analysis of the measured data obtained by the application of the piloted frequency sweep technique, characterizes the training system *total* end-to-end response, which includes the character of all elements that comprise that system. The frequency response results from visual, motion, displays, etc. can be compared directly against frequency response results produced from the equations of motion (EOM) model to determine cue-sync behavior. EOM frequency response results can then be compared directly against data produced from measurement and analysis of the aircraft in the same manner. The latter are *not* necessary for cue-sync determination.

The overall process for applying frequency domain techniques to simulator cue validation is illustrated in Figure 2. After the basic frequency domain measures have been obtained, some tolerance criteria must be applied to validate performance. Candidates for tolerance criteria are presented in this paper.

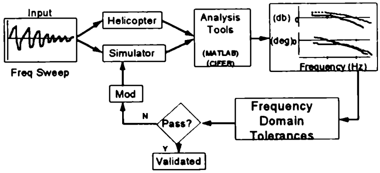


Figure 2 - Frequency Domain Validation Approach

Test Method

This desire resulted in a plan to test the MV-22A OFT using the piloted frequency sweep technique. This technique can been applied to identify many aspects of simulation character [5,9] in addition to having been applied in many areas for identification of aircraft character [3,4,6,8]. This technique allows the measurement of the cue-sync character of the training system without the need to perform any special modifications to the training system software. The training system can be measured in the same configuration in which it is used for training. Subsequent frequency response analysis of data collected utilizing the piloted frequency sweep technique can produce quantitative information about the character of the cues produced in the simulated training system in order that compliance with requirements can be easily determined. The data can also be used to assess and improve the fidelity of the simulation model where equivalent aircraft data is available.

Training System Architectural Description

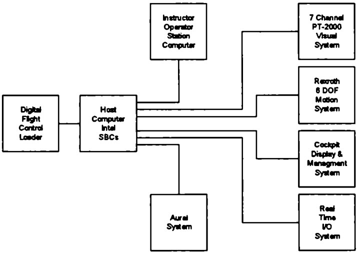

Instructor Operator Station Computer
7 Channel PT-2000 Visual System
Rexroth 6 DOF Motion System
Digital Flight Control Loader
Host Computer Intel SBCs
Cockpit Display & Managment System
Aural System
Real Time I/O System

Figure 3 - MV-22 OFT Block Diagram

Figure 3 provides a block diagram of the major components comprising the MV-22A OFT. This is a new simulator utilizing modem high performance components. The visual display system includes a 24 foot dome mounted on the six DOF motion platform.

Frequency Sweep Test Technique

A piloted frequency sweep was generated for each control axis by moving the control in a sinusoidal fashion starting at very-low frequency (20 second period -- 0.05 Hz) and continuously sweeping while increasing the sweep frequency at a linear rate up to a predetermined maximum of about 5 Hz. The pilot was coached by a data observer to ensure good frequency content. The resultant data recording is 90-100 seconds long when the sweep is completed. Figure 4 shows a sample sweep for the longitudinal axis at 200 kts. A series of three sweeps were done and subsequently concatenated (placed end-to-end) for each control axis to ensure good frequency coverage and good data recording. Measured input and output parameters for the test effort are provided in Table 1. Although the cockpit displays and instruments (Multi Function Displays, including FLIR video, and the Standby Attitude Indicator) are also important to pilot perception of the simulated aircraft, no attempt was made during this test effort to acquire data to check the character of these devices. A subsequent effort will include these devices.

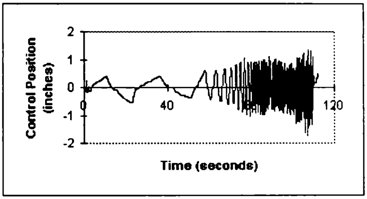


Figure 4 - Longitudinal Control Sweep (200 Kts)

The piloted frequency sweeps were performed for two flight conditions: hover and at 200 knots. The most difficult axes to sweep in the hover condition are the lateral and longitudinal cyclic control axes because the aircraft (real or simulated) tends to gain speed and is displaced from the hover condition during the long-cycle (low-frequency) sweep. The pilot is allowed to correct for this drift as long as the correction remains relatively uncorrelated with the input signal. The Throttle Control Lever (TCL) sweep was omitted in the 200 Kts flight condition. The training system software was not modified in *any* way for this test effort. Software build #164 was active in the computational system.

Table 1 - Signals Collected for Test Effort

Description	Symbol	Units
Inputs		
Directional Pedal Position	δ_{dir}	inches
Lateral Control Position	δ_{lat}	inches
Longitudinal Control Position	δ_{lon}	inches
TCL Control Position	δ_{tcl}	inches
Outputs		
Equations of Motion Roll Rate	p	rad/sec
Equations of Motion Pitch Rate	q	rad/sec
Equations of Motion Yaw Rate	r	rad/sec
Equations of Motion Load Factor	O	G
PT-2000 IG Filtered Video	n/a	volts
PT-2000 IG Unfiltered Video	n/a	volts
MotionPak X Acceleration	a_x	G
MotionPak Y Acceleration	a_y	G
MotionPak Z Acceleration	a_z	G
MotionPak Roll Rate	p	deg/sec
MotionPak Pitch Rate	q	deg/sec
MotionPak Yaw Rate	r	deg/sec

Test Equipment

Figure 5 contains a block diagram representing the equipment setup designed by the authors for data collection. A MotionPak sensor was used to measure motion platform angular rates (p, q, r) and translational accelerations (a_x, a_y, a_z). The motion sensor was mounted at a position immediately behind the pilot's seat on a rigid section of the cockpit baseframe. The mounting position is accurately known with respect to the modeled aircraft CG. Attachment was made to the Digital Flight Control Loader force actuator boards in order to obtain information on the control positions as close to the control point as could be achieved. Attachment was also made to the strip chart analog recording outputs of the real-time I/O subsystem in order to obtain the equations of motion (EOM) outputs indicated in Table 1. Finally, RGB video was obtained from the Compuscene PT-2000 Image Generator and sampled in both filtered and unfiltered format at the PC Computer. Subsequent tests on another OFT added matched low pass filters to all measured parameters to improve recorded signal quality.

The data collected were digitally sampled at 60 Hz using a commercially available data acquisition board inserted into an expansion slot of the PC Computer and custom sampling software written by the author. The data collection system currently supports 16 channels (single-ended) and also supports sampling rates up to 6 kHz per channel using all 16 channels. Higher rates are possible when the channel count is reduced. The sampled data collected was saved in ASCII columnar format for ease of delivery to other analysis software and plotting utilities.

Frequency Responses

Due to the short time limitations associated with this test effort, a MATLAB™ "m-file" (sisofra.m) previously developed by the author to perform Single-Input Single-Output (SISO) frequency response analysis was used. The m-file provides for concatenation (end-to-end placement) of both input and output time histories, scaling, linear trend and bias removal, digital filtering, tapered windowing for reduction of spectral leakage and sidelobe reduction, 50% overlapped windowing for maximum averaging, s-domain conversions (integration and/or differentiation), and SISO spectral content extraction. Multiple runs of sisofra.m can be performed in an unattended fashion. Table 2 indicates the input/output pairs used to produce the frequency responses.

Table 2 - Frequency Response Input/Output Pairs

Description	EOM	Video	MotionPak
Directional Sweep Hover & 200 kts	r_{eom}/δ_{dir}	video/δ_{dir}	r_{mpak}/δ_{dir}
Lateral Sweep Hover & 200 kts	p_{eom}/δ_{lat}	video/δ_{lat}	p_{mpak}/δ_{lat}
Longitudinal Sweep Hover & 200 kts	q_{eom}/δ_{lon}	video/δ_{lon}	q_{mpak}/δ_{lon}
Vertical Sweep Hover Only	a_z/δ_{tcl}	video/δ_{tcl}	a_{zmpak}/δ_{tcl}

Subsequent analysis efforts will make use of the more sophisticated NASA Ames Comprehensive Identification from FrEquency Responses (CIFER) product [2]. While CIFER incorporates more sophisticated analysis methodologies, CIFER setup and analysis time as compared to the MATLAB™ method used here was deemed to be too prohibitive given the very limited time available for this effort.

Frequency Response Results

After the data collection effort was complete, the data were organized in a manner convenient for analysis by the MATLAB™ m-file. The subsequent results were saved in ASCII columnar format for submission to a plotting utility. The following information was produced:

G_{xx} Input autospectrums (frequency spectra for the directional, lateral, longitudinal and throttle control lever control positions). Displayed in dB as a function of frequency.

G_{yy} Output autospectrums (frequency spectra for the roll, pitch and yaw angular rates and display video). Displayed in dB as a function of frequency.

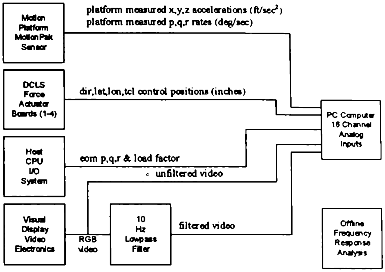
G_{xy} Input-Output cross spectrums displayed in dB as a function of frequency.

$|H_{xy}|$ Transfer function magnitude displayed in dB as a function of frequency.

$<H_{xy}$ Transfer function phase displayed in degrees as a function of frequency.

G^2_{xy} Coherence (a dimensionless measure of the response quality). Values > 0.6 are generally acceptable. Coherence represents the fraction of the output that is linearly related to the input.

The above information was produced for EOM, motion, and visual responses for each control axis for each of the flight conditions and overlaid on the same plots. A typical set of frequency response test results for the longitudinal control (pitch) axis at 200 kts is shown in Figure 6. The quality of the frequency sweeps produced for this effort were deemed to be adequate for the purposes of this paper and are based on the criteria established in the references [2,3,4].



platform measured x,y,z accelerations (ft/sec^2)
platform measured p,q,r rates (deg/sec)
Motion Platform MotionPak Sensor
DCLS Force Actuator Boards (1-4)
dir,lat,lon,tcl control positions (inches)
PC Computer 16 Channel Analog Inputs
Host CPU I/O System
eom p,q,r & load factor
unfiltered video
Visual Display Video Electronics
RGB video
10 Hz Lowpass Filter
filtered video
Offline Frequency Response Analysis

Figure 5 - Test Equipment Setup

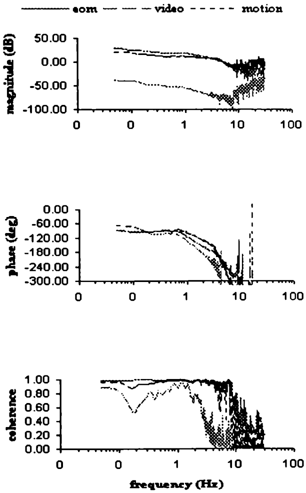


Figure 6 - Longitudinal Frequency Response (200 kts)

Additionally, spectrograms (frequency -vs- time) were produced for each input using a m-file (spectgram.m) included with the MATLAB™ Signal Processing Toolbox in order to provide quick feedback regarding the quality of the input sweeps prior to further detailed analysis. These spectrograms were utilized as a rough indication that the range of frequencies desired were adequately represented in the time histories acquired before the frequency response analysis was performed. (The desired linear change in frequency for the frequency sweeps from lower to higher values in the desired range was present

CUE VALIDATION METHODS

Once the frequency response data are available, some basis for evaluating the cue system performance must be applied. Some candidates for this basis are introduced below; two addressing the synchronization of all cues and two others addressing only motion cues.

Cue-Sync Compliance

Unnoticeable Dynamics Envelopes Research work in aircraft handling qualities has produced parameters and criteria that may be adaptable to flight simulator fidelity. The analysis of the handling qualities of aircraft with highly augmented flight control systems led to the use of equivalent systems methods. In the equivalent system method, a relatively simple transfer function is used in ground-based and in-flight simulators to characterize a very complex aircraft/high order flight control system combination which expedites the analysis process. Research data revealed that large mismatches in frequency response between the high order and equivalent low order systems were not noticed by the evaluating pilots, except when operating in the 1 to 5 rad/sec frequency range. Reference [12] describes gain and phase envelopes for maximum unnoticeable added dynamics derived from analysis of experimental data for conventional and V/STOL flight tasks. These envelopes were presented to focus the tuning process for equivalent systems applied in military standards for aircraft handling qualities. Figure 7 illustrates the envelope of maximum unnoticeable added dynamics presented in a current military standard [13]. For cue sync analysis of flight simulator fidelity, it appears that the same or similar envelopes would be appropriate. The simulator math model, motion system, and visual system can be considered equivalent systems and compared to the real world aircraft frequency response. For acceptable fidelity, the deviation of simulator gain and phase from aircraft values would be expected to fall within the gain and phase envelopes presented here. For cue-sync analysis, the motion and visual cue response difference from EOM response can be evaluated with this criteria. Of course, some experience with this criteria is necessary before definite gain and phase envelope boundaries can be declared for cue sync applications.

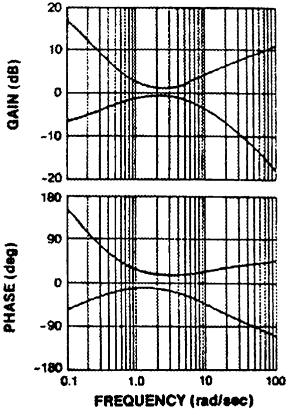


Figure 7 - Gain & Phase Envelope Plots

Bandwidth and Phase Delay Aircraft handling qualities specifications for fixed and rotary wing aircraft now utilize frequency domain parameters such as bandwidth and phase delay to characterize aircraft behavior. These same parameters can be applied to simulator fidelity and cue-sync assessment. The following method is utilized to extract bandwidth and phase delay information from the frequency responses:

Magnitude and Phase

$$magnitude = \left|\frac{X}{X_i}\right| db$$

$$(X = \Theta, \Phi, \Psi)$$

$$(X_i = F_s \, or \, \delta_s)$$

phase = f (degrees)

both versus ω(rad/sec)

Phase Delay

$$t_p = \frac{\Delta\Phi(2\omega_{180})}{57.3(2\omega_{180})}$$

Note: *If phase is nonlinear between ω_{180} and $2\omega_{180}$, τ_p is determined from a linear least squares fit to the phase curve between ω_{180} and $2\omega_{180}$.*

Bandwidth

For Rate Response types:

ω_{BW} is the lesser of ω_{BWgain} and $\omega_{BWphase}$

For Attitude Command/Attitude Hold Response (ACAH) types:

$\omega_{BW} \equiv \omega_{BWphase}$

Application of the above technique for the extraction of bandwidth and phase delay from frequency responses is used to produce the data shown in Table 3. Since only angular rates were measured during the data collection effort, an s-domain conversion (1/s integration) was applied to obtain the angular *positions* necessary for the measurement.

A missing element fundamental for characterization of EOM behavior is the aircraft frequency domain response. Discussions with flight test engineering personnel from NAWCAD indicate that the EOM character of the training system, as determined through application of classical time domain techniques, is "adequate". While equivalent aircraft responses are highly desirable, they are *not* mandatory for measurement of cue-sync. Given these limitations, the EOM frequency domain responses are *assumed* to compare favorably against the aircraft. Table 3 contains the calculated bandwidths and phase delays for the various cueing sources for the longitudinal at 200 Kts only. This step establishes the baseline from which motion and visual system response character can later be compared.

Table 3 - Bandwidths & Phase Delays for Long Axis 200 kts

Axis	Bandwidth (rad/sec)	Phase Delay (sec)
Aircraft	Not Available	Not Available
EOM	8.5	0.082
Visual	6.5	0.940
Motion	1.7	0.770

Motion Cueing Fidelity

Phase Criteria An uninvited consequence of motion cueing algorithms (due to the need to ensure that the motion system does not exceed its physical displacement limits) is that the lower frequencies are attenuated and phase shifted ahead in time. Reference [5] provides a detailed assessment of motion phase shift impact on pilot opinion for several helicopter tasks flown in the Vertical Motion Simulator at NASA Ames. This assessment applied a tolerance criteria of ±20 degree phase difference when comparing motion response to the equivalent EOM parameter. The results were summarized in terms of the frequency range where motion phase met this criteria and the cross-over frequency where the phase of each parameter was identical. Analysis of pilot handling quality ratings and other comments supported the concept that the motion cues were more satisfactory during tasks performed in frequency ranges where the ±20 degree phase tolerance was met.

Table 4 presents the results of applying the ±20 degree phase tolerance to the frequency response results for motion cueing quality assessment. These results are a quantitative measure of how well the respective motion axes matched the corresponding EOM axes. Although not performed for this test effort, a rating mechanism such as this should be further substantiated with pilot evaluation.

Table 4 - Motion Cueing Fidelity Range

Axis	Lower Freq f_1 (Hz)	Upper Freq f_2 (Hz)	Crossover Freq f_c (Hz)	"Hi-Fi" Range
Dir- Hover	0.180	2.600	1.000	2.420
Dir- 200kts	0.160	2.200	0.850	2.045
Lat- Hover	0.190	4.700	2.900	4.510
Lat- 200kts	0.100	3.500	1.800	3.400
Lon- Hover	0.120	5.000	3.000 *	4.880
Lon- 200kts	**	**	3.6 **	**

** - phase always within tolerance

FAA Alternative Method The FAA has issuedAdvisory Circular (AC) 120-63 [14] to establish a way to certify helicopter simulators for use in training programs. This AC makes a general recommendation that frequency response data be utilized when available for comparing helicopter and simulator dynamic response characteristics. Appendix 2 of the AC describes two methods for motion system testing: one that documents time history response for later repeatability comparison, and another method that documents frequency response of the end-to-end motion system. The frequency response method is conducted as follows: At the point at which the accelerations from the equation normally excite the motion system, including the washout algorithms, a sinusoidal input would be used to excite the motion system. Acceleration at the pilot station would be measured to determine frequency response. The resulting frequency response measured in each axis must comply with the following specification: Gain tolerance: ±2 dB from 0.5 Hz to 5.0 Hz; Phase tolerance 0 - ±20 degrees from 1.0 Hz to 2.0 Hz. These tolerances were derived by consensus of simulator experts during the AC development process; notice that the frequency range is limited to the area where pilots typically conduct aggressive flight tasks and that the frequency ranges are different for gain and phase.

Summary of Proposed Methods

Simulator frequency response test techniques and data analysis methods for cue validation have been described in this paper. Frequency domain test results can be interpreted in a variety of ways that are potentially useful to flight simulator cue-sync analysis. Simulator cue fidelity can be accurately determined if aircraft frequency response data is available as reference criteria. Even without aircraft data, the EOM response can be utilized as criteria for quantifiable analysis of cue-sync character of all cueing systems present.

Cue-Sync Compliance Characterization

Two analysis approaches are described, both derived from aircraft handling qualities technology.

- Bandwidth and Phase Delay offers the potential for direct comparison of simulator cues with aircraft flight test data. Further evaluation of this approach must await the more widespread use of frequency domain techniques by the flight test community.

- Unnoticeable Dynamics Envelopes offer a straight forward tolerance criteria for assessing the difference between cues and when available, the aircraft, over a broad frequency range. Further experience is needed to confirm the exact envelope values for simulator application.

Motion Cueing Fidelity

Evaluation of motion cues has traditionally been a subjective process. Two methods are described in this paper to establish a more quantitative basis for assessing motion cue fidelity.

- Fidelity Range Determination is based on the simple criteria that motion phase remain within 20 degrees of the input signal. Further experience is needed to clarify the interpretation of the resulting identified "hi-fi" frequency range for typical simulator tasks.

- FAA Alternative Method imposes straight forward tolerance on motion gain and phase response over specific frequency ranges. Experience with this method is needed to see if these tolerance values are effective.

Conclusion

While a tolerance criteria for comparison against requirements has not been quantified, it has been characterized. New techniques are suggested that differ from previous methods. A means to quantitatively measure cue-sync character without the necessity for modification of training system software has been demonstrated.

Recommendations

- The piloted frequency sweep technique can be used for numerous purposes, all or many of which can have a positive impact on the quality and fidelity of simulated rotary wing flight training systems. Additional test and/or research efforts should be initiated to further substantiate use of the piloted frequency sweep technique as the primary means of determination of cue-sync character.

- Further efforts are necessary to establish a quantitative tolerance criteria for comparison of frequency response results for EOM performance against the aircraft. Similar efforts are also necessary for quantifying cueing tolerance relationships to EOM behavior with the realization of what is practical within the physical limitation of the cueing systems.

- While the MATLAB™ m-files developed for use during the limited time available for this test effort were deemed adequate (and potentially quite useful for *rapid* frequency response estimates), the frequency sweep data from this effort should be re-processed using the CIFER (Comprehensive Identification from FrEquency Responses) utility. This utility contains several programs for the analysis of frequency response data. The subprograms FRESPID (Frequency RESPonse IDentification), MISOSA (Multi-Input Single-Output Spectral Analysis), and COMPOSITE (Multi Window Averaging) all serve for production of wide band, high fidelity frequency responses. The additional subprograms DERIVID (Generalized Stability Derivative Identification from Frequency Responses), VERIFY (State Space Model Verification), and NAVFIT (General Purpose Frequency Response Fitting) work together for identification and evaluation/verification of state-space models of up to 40 states.

- Development of a method to incorporate real-time spectrograms for use in the cockpit while applying the frequency sweep technique. Real-time spectrograms can provide immediate feedback to the pilot on the quality of the sweep as it is being generated in order to ensure adequate data is obtained prior to detailed analysis.

- Development of a technique for measurement of cue-sync character of the cockpit displays and related devices. These devices are equally as important to pilot perception of the simulated aircraft as any of the other major subsystems measured for this effort. One interesting approach to measurement of video based instrument displays is described in reference [7].

REFERENCES

[1] Butrimas, S.; Browder, B.: *A Unique Approach To Specification and Testing of Simulators*, AIAA 87-2570-CP, Aug 1987.

[2] Tischler, M. B.; and Cauffman, M. G.: *CIFER Version 2.1: Comprehensive Identification from FrEquency Responses - An Interactive facility for system identification and verification, Descriptive Overview*, Undated.

[3] Tischler, M. B.: *Frequency-Response Identification of XV-15 Tilt-Rotor Aircraft Dynamics*, NASA TM-89428/USAAVSCOM TM 87-A-2, May 1987.

[4] Tischler, M. B.; and Cauffman, M. G.: *Frequency-Response Method for Rotorcraft System Identification with Applications to the BO-105 Helicopter*, Presented at the 46th Annual Forum of the American Helicopter Society, Washington, D.C., May 1990.

[5] Atencio, A.: *Fidelity Assessment of a UH-60A Simulation on the NASA AMES Vertical Motion Simulator*," NASA TM-104016/USAATCOM TR 93-A-005, Sep 1993.

[6] Tischler, M. B.; and Fletcher, J. W.; et al: *Demonstration of Frequency Sweep Testing Technique Using a Bell 214-ST Helicopter*, NASA TM-89422/USAAVSCOM TM-87-A-1, April 1987.

[7] Bryant, R. B.; et al: *Dynamic Latency Measurement Using The Simulator Network Analysis Project (SNAP)*, Proceedings of the 16th Interservice/Industry Training Systems and Education Conference, Nov 1994.

[8] Hamm, J. A., Gardner, C.K., Tischler, M. B., *Flight Testing and Frequency Domain Analysis for Rotorcraft Handling Qualities Characteristics*, AHS Specialists' Meeting on Piloting Vertical Flight Aircraft, 20-23 Jan 1993, San Francisco, CA.

[9] Schroeder, J. A., Watson, D. C., Tischler M. B., Eshow, M. M.: *Identification and Simulation Evaluation of an AH-64 Helicopter Math Model*, AIAA Atmospheric Flight Mechanics Conference, 12-14 Aug 1991, New Orleans, LA.

[10] Bray, R. S.: *Visual and Motion Cueing in Helicopter Simulation*, NASA TM-86818, Sep 1985.

[11] *"MATLAB™" User's Guide* (Version 4.2), The MATH Works Inc., Natick, MA , Jul 1994.

[12] Hodgkinson, J.; and Wood, J. R.: "Equivalent Systems Criteria for the Flying Qualities Mil Standard" Presented at the Joint Flying Qualities/Flight Controls Symposium Sponsored by AFWAL/FDL, Wright Patterson Air Force Base, 2-5 March 1982.

[13] MIL-STD-1797A, *Flying Qualities of Piloted Aircraft*, 30 January, 1990.

[14] AC-120-63, *Helicopter Simulator Qualification*, U.S. Department of Transportation, Federal Aviation Administration. Dated: 10/11/94.

AIAA-96-3529-CP

STATUS OF A COMPREHENSIVE VALIDATION OF THE NAVY'S F/A-18A/B/C/D AERODYNAMICS MODEL

Michael S. Bonner
Naval Air Warfare Center Aircraft Division
Flight Vehicle Simulation Branch
MS-3, 48140 Standley Rd.
Patuxent River, 20670-5304
(301)342-7601 FAX:(301)342-7607

David R. Gingras*
Science Applications International Corporation
Systems Technology Group
44417 Pecan Court, Suite B
California, MD 20619-3272 USA
(301)863-5077 FAX:(301)863-0299

Abstract

A flight test program was designed and flown with two United States Navy F/A-18 aircraft at the Naval Air Warfare Center, Patuxent River, MD to collect data for validation of the Manned Flight Simulator's F/A-18 aerodynamics model. This paper details the flight test program and the processes used to calibrate the flight data for the creation of a truth-data set to be used for the validation as well as source for future updates. The paper presents examples of preliminary aerodynamic coefficient comparisons between model predicted values and values extracted from flight. It also provides a discussion of a coefficient comparison acceptance criteria currently being developed.

Nomenclature

Symbols

A_x	Longitudinal acceleration, (ft/s^2)
A_y	Lateral acceleration, (ft/s^2)
A_z	Normal acceleration, (ft/s^2)
b	Wing span, (ft)
c	Mean aerodynamic chord, (ft)
C_D	Drag coefficient
C_l	Rolling moment coefficient.
C_L	Lift coefficient
C_m	Pitch moment coefficient
C_n	Yawing moment coefficient
C_x	Axial force coefficient.
C_y	Lateral force coefficient
C_z	Normal force coefficient
F	Force, (lbs)
g	Gravity, (32.174 ft/s^2)
I_{xx}	Moment of inertia about X- axis — body frame, (lbs-ft^2)
I_{yy}	Moment of inertia about Y- axis — body frame, (lbs-ft^2)
I_{zz}	Moment of inertia about Z- axis — body frame, (lbs-ft^2)
I_{xy},I_{xz},I_{yz}	Products of inertia, (lbs-ft^2)
m	Mass, (slugs).
M	Moment, (ft-lbs)
N	Number of samples
p	Roll rate, (rad/s)
q	Pitch rate, (rad/s)
$\bar{q}$	Dynamic pressure (psf)
r	Process output or yaw rate, (rad/s)
S	Reference planform area, (ft^2)
u	X-axis velocity — body frame, (ft/s)
U	Theil Inequality Coefficient
v	Y-axis velocity — body frame, (ft/s)
V_{True}	Velocity, (ft/s)
w	Z-axis velocity — body frame, (ft/s)
$W_{C...}$	Weighting factor
X,Y	Arbitrary parameters
α	Angle of attack, (rad)
β	Angle of sideslip, (rad)
ϕ	Roll angle, (rad)
θ	Pitch angle, (rad)
ψ	Yaw angle, (rad)
ρ	Correlation coefficient
σ	Standard deviation

[*] Aerospace Engineer, Member AIAA

Subscripts

m	measured
i,j,k	Indices.

Superscripts

^	Estimated
~	Derived quantity
—	Mean value

Introduction

The Manned Flight Simulator (MFS) is responsible for providing real-time and non-real-time simulation support for the test and evaluation of naval aircraft[1]. As part of this task, the MFS has maintained high-fidelity 6 degree-of-freedom (DOF) simulations of the F/A-18 A/B/C/D aircraft, including support for several common ordnance loadings. The original F/A-18 A/B simulation was received from McDonnell-Douglas Aerospace (MDA) in 1983 and was adapted to fit the MFS standard simulation architecture known as the Controls Analysis and Simulation Test Loop Environment (CASTLE). Since then, the Navy has made a considerable effort to improve the F/A-18 simulation aerodynamics model using wind-tunnel data, parameter identification (PID), and manual adjustment based on pilot flying qualities assessments[2,3,4,5].

Many components of the F/A-18 aerodynamics model have been compared to wind-tunnel data as well as flight data[3,4,6]. The efforts have served well for the verification, and in some cases validation, of the aerodynamics model at various stages of its development.

The main goal of this effort is to perform a comprehensive evaluation of the F/A-18 aerodynamic model's capability to accurately predict the aerodynamic coefficients using flight data collected from test flights spanning the normal flight envelope as defined in the United States (U.S.) Navy F/A-18 NATOPS flight manual[7]. A secondary goal is to develop a well-documented "truth" data set to be used to evaluate potential modifications to the model.

The diagram in Fig. 1 outlines the process used to validate the F/A-18 aerodynamics model. This paper details each step of the process while providing insight into the problems encountered while analyzing data collected with production aircraft sensors. The paper will also present an example of preliminary aerodynamic coefficient comparisons and results as well as discuss plans for future work.

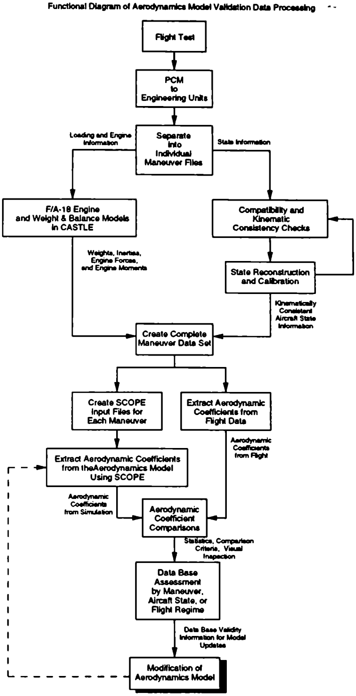

Functional Diagram of Aerodynamics Model Validation Data Processing
Flight Test
PCM to Engineering Units
Separate into Individual Maneuver Files
Loading and Engine Information
State Information
F/A-18 Engine and Weight & Balance Models in CASTLE
Compatibility and Kinematic Consistency Checks
Weights, Inertias, Engine Forces, and Engine Moments
State Reconstruction and Calibration
Kinematically Consistent Aircraft State Information
Create Complete Maneuver Data Set
Create SCOPE Input Files for Each Maneuver
Extract Aerodynamic Coefficients from Flight Data
Extract Aerodynamic Coefficients from theAerodynamics Model Using SCOPE
Aerodynamic Coefficients from Flight
Aerodynamic Coefficients from Simulation
Aerodynamic Coefficient Comparisons
Statistics, Comparison Criteria, Visual Inspection
Data Base Assessment by Maneuver, Aircaft State, or Flight Regime
Data Base Validity Information for Model Updates
Modification of Aerodynamics Model

Fig. 1 Validation process diagram.

Flight Tests

Data from over 2000 individual maneuvers were collected with two aircraft in five different loading configurations during 33 test flights consisting of over 100 hours of flight time using F/A-18A BuNo 161744 (SD-102), a Lot V single-seat aircraft, and F/A-18D BuNo 163986 (SD-113), a Lot XII two-seat aircraft (Fig. 2).

Instrumentation

Both aircraft were equipped with production instrumentation as well as a Quick Installation Data System (QUIDS) package to record variables from the 1553 MUX bus on magnetic tape. Because of time and budget constraints, production aircraft sensors were used

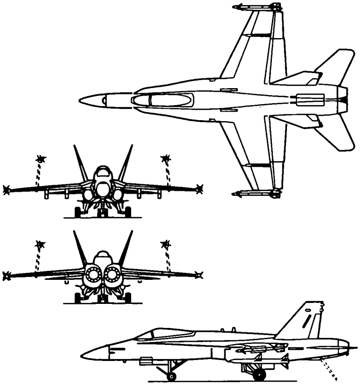


Fig. 2 Four-view diagram of the F/A-18 C.

to measure all variables with the exception of fuel quantities in SD-102. Fuel sensors were added to SD-102 because they were unavailable on the MUX bus. Multiple redundant variables were recorded for backup and kinematic consistency checking and calibration.

The F/A-18 contains both redundant Flight Control Set (FCS) and strap-down Inertial Navigation Set (INS) angular-rate and linear acceleration sensors. The SD-102 test aircraft contained an ASN-130A INS which uses mechanical angular-rate gyroscopes, while SD-113 contained an ASN-139, which uses ring-laser gyroscopes.

The F/A-18 also has an Air-Data Computer (ADC) that provides the Mission Computer (MC) with ambient and total temperatures, local and true angle of attack, static and impact pressure, mach number, true and indicated airspeed, pressure altitude, and magnetic heading. Information is provided to the ADC by a total-temperature probe, a nose mounted pitot probe, and dual angle-of-attack vanes mounted on the lower fore-body of the fuselage.

Loadings

Data were collected with five of the six basic aircraft loadings. The single-seat F/A-18 was flown in the fighter-escort (FE), fighter-escort with center-line tank (FCL), and interdiction loadings (INT). Because of time constraints only FE and FCL loadings were flown with the two-seat aircraft. Figure 3 contains a diagram detailing each of the loading configurations. Weight and balance information was computed using the MFS F/A-18 simulation weight and balance model. Details concerning weight and balance information for each loading are presented in Table I.

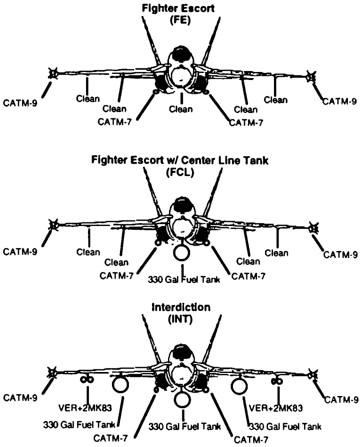


Fig. 3 F/A-18 test loadings.

Test Matrix

The test matrix was designed to cover the standard flight envelope as defined in the F/A-18 Naval Aviation Training and Operational Procedures Manual (NATOPS). Integrated test blocks were flown in level 1G flight conditions for the indicated Mach number. Additional maneuvers were performed in accelerated conditions at 35,000 ft altitude. Powered Approach, full- (PA) and half-flap (PA1/2), data were collected at 6°, 8.1° (on-speed), 10°, and 12° AOA at 5,000 ft altitude. Figure 4 contains a graphic of the single-seat test envelope. The two-seat test envelope is shown in Fig. 5.

Table I contains a break down of the maneuvers flown with the three-axis control augmentation system on (CAS). Additional test points were flown with CAS only in the pitch axis to collect data for future parameter identification efforts. Integrated Test Blocks (ITB) containing maneuvers such as 3211s, various rolls (180°, bank-to-bank, etc.), and steady heading sideslips were designed to stimulate motion in each of the three main axes at each flight condition[8]. Additional maneuvers were flown to gather wind-condition and aircraft performance information.

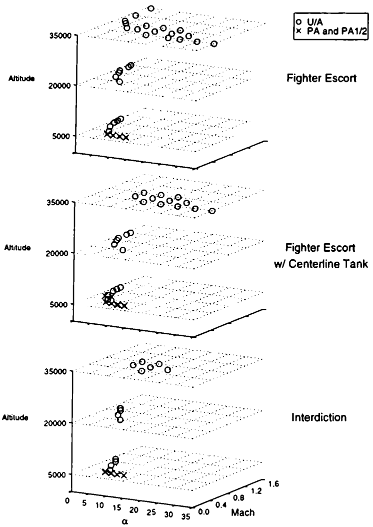


Fig. 4 F/A-18 A test envelope.

Table I. Maneuvers flown and analyzed for .the validation effort.

Model	Loading	Maneuvers Flown	Maneuvers Analyzed
A	FE	462	190
	FCL	317	155
	INT	261	110
D	FE	320	134
	FCL	75	51
TOTAL		1435	640

During test plan development prior to the flight test, many of the ITBs were flown in the MFS F/A-18 real-time simulator by the test pilots. After pre-flying test points, the pilots were able to provide valuable input to the formation of a final flight plan as well as take note of potentially hazardous areas of the test matrix. This also provided pilot training for some of the non-standard maneuvers used in this project.

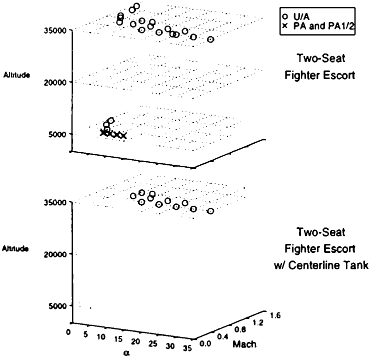


Fig. 5 F/A-18 D test envelope.

Validation Data Set

One of the main objectives in creating the validation data set was to establish a set of calibrated, kinematically consistent data. The flight data were analyzed for consistency among redundant sources, then were verified for kinematic consistency. In cases where data were not consistent, calibrations were applied and the data rechecked for consistency. The following sections present results of this process.

Redundant Measurement Consistency Check

A redundant-measurement flight-data consistency check was performed to assess the quality of the measured flight data. During this process, redundant variables recorded by the INS, the Flight Control Electronics Set (FCS), and the ADC sensors were compared to one another. Redundant sensors were available for angular rates and linear accelerations.

FCS/INS Sensors

Measurements from FCS and INS sensors were consistent with one another in most cases for both SD-102 and SD-113. For both aircraft, dynamic matching between FCS and INS angular-rate sensors was excellent. Figure 6 contains a typical comparison of FCS angular rates to the INS wide-band rates for a pitch 3211 maneuver performed by SD-102.

During data reduction linear accelerations measured by the FCS and INS had been recomputed at the aircraft center of gravity using sensor locations noted in weight and balance reports. A problem was seen in comparisons of lateral acceleration signals during high roll rate maneuvers performed by both aircraft. Large discrepancies, such as 6 ft/sec^2 errors in lateral acceleration during a 200+ deg/sec roll rate, were seen in the single-seat data. Using parameter identification techniques [9], an error in the INS lateral sensor position on the order of 0.5 in was estimated. Further research

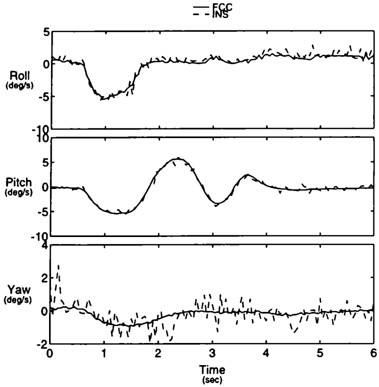


Fig. 6 Wide-band INS angular rate measurements compared to FCS rate measurements.

lead to a McDonnell Douglas Aerospace internal memo noting a "center of navigation" position for the INS that validated the position error estimate[10].

Comparisons between INS and FCS signals also revealed a time delay between the two for both aircraft. Wide-band INS signals were in sync with FCS quantities, while narrow-band INS signals were delayed by approximately 0.1 to 0.2 seconds (Fig. 7). The dynamics represented in the signals were almost identical.

A decision was made to use INS signals for several reasons. First, during the analysis of SD-102 data it was found that FCS rate gyros would sometimes stop updating for one frame during high-rate maneuvers. This phenomenon was seen randomly throughout maneuvers and could not be correlated with any variables. Next, only normal and lateral accelerations were available from the FCS sensors, therefore the INS axial accelerations *had* to be used. And last, delays of the same magnitude as those of the INS narrow-band signal were found in the recorded control surface positions[11]. The use of aircraft information with inconsistent time tags as input to the aerodynamics model to extract aerodynamic coefficients can lead to erroneous coefficient comparisons.

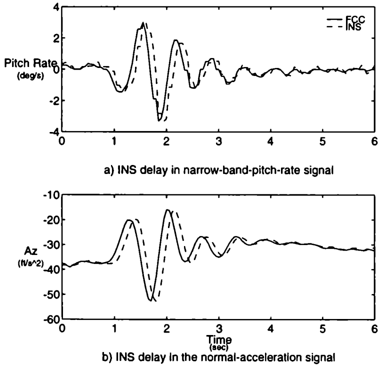
ADC/INS Sensors

Several comparisons were made between redundant angle of attack measurements for a variety of test points. The majority of the comparisons matched dynamically, but were biased from each other (Fig. 8a). At angles of attack exceeding 15°, consistency between the INS and ADC was lost (Fig. 8b).

Flight Data Kinematic Consistency

To ensure the compatibility of the measured flight data, a kinematic consistency analysis was performed. The objective of the analysis was to ensure the measured aircraft state was consistent with its corresponding state reconstructed by integrating the equations of motion.

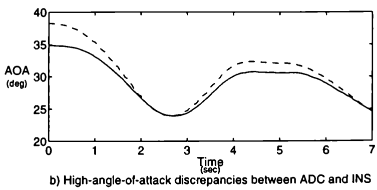
The equations used in the analysis were the six-degree-of-freedom-kinematic equations of motion



a) INS delay in narrow-band-pitch-rate signal

b) INS delay in the normal-acceleration signal

Fig. 7 INS and FCS measurement comparisons showing a delay in INS signals.



a) Low-angle-of-attack discrepancies between ADC and INS

b) High-angle-of-attack discrepancies between ADC and INS

Fig. 8 High- and low-angle-of-attack ADC and INS angle of attack comparisons.

assuming a non-rotating earth and constant winds[12,9,13]. (equation set 1). Aircraft velocity components and attitude were computed using a second-order-open Adams-Bashforth integration algorithm with INS measured linear accelerations and angular rates as inputs.

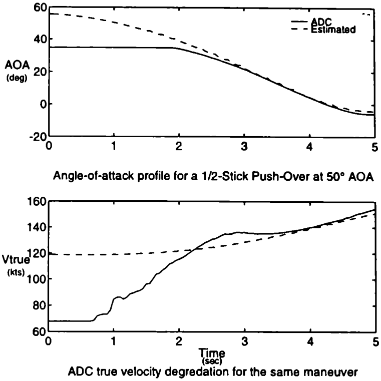
$$
\begin{aligned}
\dot{\hat{u}}_i &= A_{x_i} - g\sin\hat{\theta}_i - q_i\hat{w}_i + r_i\hat{v}_i \\
\dot{\hat{v}}_i &= A_{y_i} + g\sin\hat{\phi}_i\cos\hat{\theta}_i - r_i\hat{u}_i + p_i\hat{w}_i \\
\dot{\hat{w}}_i &= A_{z_i} + g\cos\hat{\phi}_i\cos\hat{\theta}_i - p_i\hat{v}_i + q_i\hat{u}_i \\
\dot{\hat{\psi}}_i &= (q_i\sin\hat{\phi}_i + r_i\cos\hat{\phi}_i)/\cos\hat{\theta}_i \qquad (1) \\
\dot{\hat{\theta}}_i &= q_i\cos\hat{\phi}_i - r_i\sin\hat{\phi}_i \\
\dot{\hat{\phi}}_i &= p_i + (q_i\sin\hat{\phi}_i + r_i\cos\hat{\phi}_i)\tan\hat{\theta}_i \\
\dot{\hat{h}}_i &= \hat{u}_i\sin\hat{\theta}_i - \hat{v}_i\sin\hat{\phi}_i\cos\hat{\theta}_i - \hat{w}_i\cos\hat{\phi}_i\cos\hat{\theta}_i
\end{aligned}
$$

Angular accelerations used in the equations were determined using a second-order-central-difference numerical algorithm.

The chosen observation variables included the aircraft attitude angles, $\hat{\phi}$, $\hat{\theta}$, $\hat{\psi}, h$, and the aircraft wind-relative velocity and flow angles, $\hat{V}_{true}$, $\hat{\alpha}$, $\hat{\beta}$ (equation set 2).

$$
\begin{aligned}
\hat{V}_{true} &= \sqrt{\hat{u}^2 + \hat{v}^2 + \hat{w}^2} \\
\hat{\alpha} &= \tan^{-1}\left(\frac{\hat{w}}{\hat{u}}\right) \qquad (2) \\
\hat{\beta} &= \sin^{-1}\left(\frac{\hat{v}}{\hat{V}}\right)
\end{aligned}
$$

Preliminary results showed very consistent aircraft attitude measurements from the INS, but revealed inconsistencies in the ADC velocity and angle-of-attack measurements. Airspeed and angle of attack data quality degraded as angles of attack increased above 15° for Mach numbers less than 0.6. An example of this is shown in data from a 50° AOA push over in Fig. 9. This inconsistency was due to the excessive local angles of attack in the vicinity of the fuselage mounted probes. The production angle-of-attack probes in the F/A-18 are mechanically limited at 56° local angle of attack. This corresponds to approximately 35° true angle of attack, therefore angle-of-attack information above 35° true was lost. Inconsistencies in true airspeed were also found at Mach numbers greater than 0.6 above 10° angle of attack. Analyses of the ADC true velocity signals indicated decreases in airspeed which were not consistent with angle of attack or measured accelerations (Fig. 10).



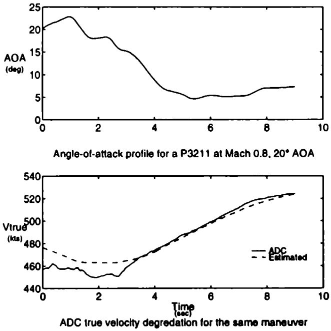
Angle-of-attack profile for a 1/2-Stick Push-Over at 50° AOA

ADC true velocity degredation for the same maneuver

Fig. 9 Production angle-of-attack sensor probe limitations and ADC true velocity degradations at high angles of attack.

Flight Data Calibration and Reconstruction

The main objective in the calibration of the flight data was to create kinematically consistent data by correcting or reconstructing data that may have been corrupted by measurement error. Calibrations and reconstructions were developed by using redundant sensors, enforcing kinematic consistency in the data, and deriving calibration information from alternate data sets.



Angle-of-attack profile for a P3211 at Mach 0.8, 20° AOA

ADC true velocity degredation for the same maneuver

Fig. 10 Example of ADC reported true velocity inconsistencies for Mach numbers greater than 0.6 at moderate angles of attack.

True Velocity and Angle of Attack

Air data for high-angle-of-attack maneuvers were reconstructed by combining the short-term accuracy of the INS with the long-term accuracy of the ADC. This approach was a compromise between using ADC data that were poor at high angles of attack and using INS data that seemed to be biased at low angles of attack. The air-data reconstruction was based on the assumptions of constant winds, negligible vertical wind component, and accurate MC reported wind measurements below 25° AOA. Wind-relative aircraft velocities were computed in the inertial frame of reference by subtracting mission computer reported winds from INS report inertial velocities. The wind relative velocities were then transferred from the inertial frame to the body reference frame where they were used to compute true airspeed and angle of attack.

This method was applied to all test points regardless of the flight condition to maintain a common data source among all air data. An example of the results from the air data reconstruction is presented in Fig. 10.

Angle of Sideslip

Preliminary comparisons of estimated angle of sideslip and INS sideslip indicated drift and bias errors. The F/A-18 has no production sensor to directly measure angle of sideslip (AOS). The variable is calculated by the mission computer using fixed wind estimates as illustrated in Fig. 11. Small changes in atmospheric conditions can lead to sideslip bias and drift errors. To maintain reasonable sideslip measurements, the sideslip signal was initialized before each maneuver or set of maneuvers. These resets were performed by the pilot initiating a HUD reject 2. This action caused the MC to re-calculate the wind estimates assuming zero angle of sideslip. These estimates were frozen for the computation of sideslip.

While initiating a HUD reject 2 the pilot was to maintain steady, wings level, 1g trim flight, without sideslip for at least 10 seconds. Given that neither of the test aircraft were equipped with a sideslip indicator, such as a yaw string or chin vane, the pilot was left to determine when the required condition was met. Human beings, by nature, are unreliable measurement devices[14], therefore random bias errors were introduced into sideslip signal.

The sideslip bias errors were particularly apparent in several cases where preliminary comparisons of simulation predicted lateral-directional coefficients to reconstructed coefficients showed large constant biases which were not consistent with the aircraft state and control surface positions. Small biases were applied to the AOS signal used to drive the simulation, thus improving comparisons of trim values of coefficient comparisons (Fig. 12). Further evidence of angle-of-sideslip biases can be seen by the scatter of circle symbols in Fig 13. The figure contains a plot of INS estimated sideslip for two-seat FE maneuvers against lateral acceleration during the initial trim portion of each maneuver. During this portion of the maneuvers, a linear relationship between sideslip and lateral acceleration was expected.

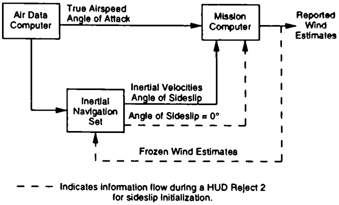
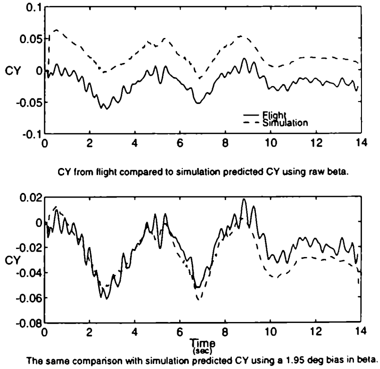


Fig. 11 Functional diagram of mission computer angle of sideslip determination

Several different methods were used in the attempt to calibrate angle of sideslip data. Among them were traditional parameter identification techniques[9], where attempts were made to identify biases in sideslip using a maximum likelihood scheme. A second attempt involved using differential angle-of-attack signals to compute sideslip[11]. Results from this method showed no overall improvement.

The calibration technique which yielded the best



CY from flight compared to simulation predicted CY using raw beta.

The same comparison with simulation predicted CY using a 1.95 deg bias in beta.

Fig. 12 Side force coefficient comparison showing result of sideslip bias.

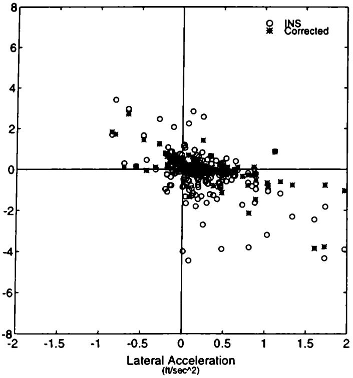


Fig. 13 Scatter plot of sideslip angle versus lateral acceleration during trim showing results of the side slip calibration.

results was rather unconventional. An assumption was made that the F/A-18 aerodynamic model accurately predicted lateral-direction coefficients during steady-state, unaccelerated trim conditions. The F/A-18 aerodynamic model inputs were overridden with kinematically consistent aircraft state information (including INS estimated sideslip angle) and measured control surface deflections. Values of C_l, C_n, and C_Y were predicted by the F/A-18 aerodynamics model using an analysis tool called Simulation Checking using an Optimal Prediction Evaluation (SCOPE). These coefficients were compared with corresponding coefficients extracted from flight data for the trim portion of each maneuver. Differences between the coefficients similar to the example shown in Fig. 12, were assumed to be a result of sideslip. Next, using the linear model extraction (LME) tool in CASTLE, values of $C_{l\beta}$, $C_{n\beta}$, and $C_{Y\beta}$ were computed for each data ITB trim condition. Sideslip corrections were determined by combining the contribution of each coefficient comparison and LME derivative using a weighted average as defined in equation set (3).

$$\Delta\beta = W_{C_Y}\Delta\beta_{C_Y} + W_{C_l}\Delta\beta_{C_l} + W_{C_n}\Delta\beta_{C_n}$$

Where Wc_Y, $\Delta\beta c_Y$, Wc_l, $\Delta\beta c_l$, Wc_n and $\Delta\beta c_n$ were determined in the following fashion.

$$W_{C_Y} = \frac{C_{Y_\beta}}{C_{Y_\beta} + C_{l_\beta} + C_{n_\beta}} \qquad (3)$$

$$\Delta\beta_{C_Y} = \frac{\Delta C_Y}{C_{Y_\beta}}$$

This process resulted in considerable corrections in sideslip for many maneuvers, but was not effective for maneuvers where trim data were missing. Figure 13 contains a plot of two-seat FE angle of sideslip values versus lateral acceleration during trim for corrected sideslip values (star-shaped symbols) in addition to uncorrected values (circle-shaped symbols). It is easy to notice the improvement in angle-of-sideslip values especially in the region where the aircraft was experiencing little or no lateral acceleration.

The complete calibrated observation parameter set was compared to estimated quantities computed using equation set (1) and (2). Figure 14 presents an example of kinematic data consistency. In general kinematic consistency was very good among the data. Some drift was encountered in several of the estimated observation parameters V_{True}, α, β, φ, θ, and ψ. (3) Figure 14 also contains a statistical measure of the quality of the comparison called the Theil inequality coefficient[15], equation set (4). This parameter is the

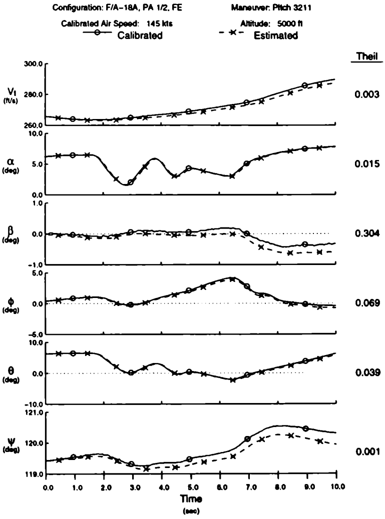


Fig. 14 Example of kinematic consistency.

root mean squared (RMS) error normalized by the sum of the RMS values of the signals being compared. The coefficient ranges from 0, where there are no differences between the two signals, to 1, where the RMS error between the two signals is equal to the sum of the RMS of the two signals.

$$U = \frac{\sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(Y_i - \hat{Y}_i\right)^2}}{\sqrt{\frac{1}{N}\sum_{i=1}^{N}Y_i^2} + \sqrt{\frac{1}{N}\sum_{i=1}^{N}\hat{Y}_i^2}} \tag{4}$$

In addition to the Theil coefficient, error proportions were computed for each maneuver (equation set 5).

$$\begin{aligned} U_{Bias} &= \frac{\left(\overline{Y} - \overline{\hat{Y}}\right)^2}{\frac{1}{N}\sum_{i=1}^{N}\left(Y_i - \hat{Y}_i\right)^2} \\ U_{Var} &= \frac{\left(\sigma - \hat{\sigma}\right)^2}{\frac{1}{N}\sum_{i=1}^{N}\left(Y_i - \hat{Y}_i\right)^2} \\ U_{Cov} &= \frac{\left(2(1-\rho)\sigma\hat{\sigma}\right)^2}{\frac{1}{N}\sum_{i=1}^{N}\left(Y_i - \hat{Y}_i\right)^2} \end{aligned} \tag{5}$$

where

$$\overline{Y} = \frac{1}{N}\sum_{i=1}^{N}Y_i$$

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(Y_i - \overline{Y}\right)^2}$$

$$\rho = \frac{\frac{1}{N}\sum_{i=1}^{N}\left(Y_i - \overline{Y}\right)\left(\hat{Y}_i - \overline{\hat{Y}}_i\right)}{\sigma\hat{\sigma}}$$

These statistics, ranging between 0 and 1, indicate sources of differences between two signals. Large values of the covariance proportion indicated differences due to uncorrelated factors such as noise. Large variance proportion values generally indicated scaling errors. Large bias proportion values indicated drift errors. Plots of Theil statistics versus angle of attack and Mach aided the engineers in assessing data quality for the entire 640 maneuvers.

Weight, Balance and Thrust Reconstruction

Prior to the aerodynamic coefficient reconstruction, SCOPE was used to drive the F/A-18 simulation to extract weight, inertial, and center of gravity location predictions from the weight and balance model. Baseline aircraft weights and longitudinal center of gravity (CG) positions were taken from actual measurements of the instrumented test articles performed prior to execution of the flight test program. Fuel-tank quantities were measured during the test and used in the weight and balance model. The simulation model was successfully validated with aircraft weight and balance information from a manufacturer's weight and balance report[16] as well as preflight weight and balance measurements. A detailed uncertainty analysis performed for the data reduction process indicated that fuel weight measurement errors had negligible effects on data reduction results.

Because of time and budget constraints, the test articles' engines were not thoroughly instrumented and calibrated. Thrust estimates were extracted from the F/A-18 simulation engine model for each maneuver condition and throttle setting. The engine model consisted of steady-state engine performance data derived from a non-real-time cycle deck. Static values of net thrust, fuel flow, and related engine operating variables were determined via table lookup based on altitude, ambient temperature, Mach, and throttle position. Second-order filters were used to model dynamic engine response.

Aerodynamic Data Reconstruction

Aerodynamic coefficients were reconstructed from the calibrated data. Rates and accelerations were used with the aircraft mass and inertia characteristics to estimate the total forces and moments acting on the aircraft. From them, forces and moments due to thrust were subtracted, yielding aerodynamic contributions. The aerodynamic forces and moments were nondimensionalized to provide the aerodynamic coefficients of the aircraft. The longitudinal coefficients were transferred from the body axis to the stability axis and all moment coefficient reference points were transferred from the actual center of gravity to the reference location at 25% mean aerodynamic chord for comparison with the simulation. See equation set (6).

$$
\begin{aligned}
C_x &= \frac{m}{\bar{q}S}\left(F_x - F_{x_{Eng}}\right) \\
C_y &= \frac{m}{\bar{q}S}\left(F_y - F_{y_{Eng}}\right) \\
C_z &= \frac{m}{\bar{q}S}\left(F_z - F_{z_{Eng}}\right) \\
C_l &= \frac{m}{\bar{q}Sb}\left(M_x - M_{x_{Eng}} - F_y \Delta z_{cg} + F_z \Delta y_{cg}\right) \\
C_m &= \frac{m}{\bar{q}S\bar{c}}\left(M_y - M_{y_{Eng}} + F_x \Delta z_{cg} - F_z \Delta x_{cg}\right) \\
C_n &= \frac{m}{\bar{q}Sb}\left(M_z - M_{z_{Eng}} - F_x \Delta y_{cg} + F_y \Delta x_{cg}\right)
\end{aligned}
\tag{6}
$$

with

$$
\begin{aligned}
C_L &= C_x \sin\alpha - C_z \cos\alpha \\
C_D &= -C_x \cos\alpha - C_z \sin\alpha
\end{aligned}
$$

where

$$
\begin{aligned}
F_X &= mA_x \\
F_Y &= mA_y \\
F_Z &= mA_z \\
M_x &= \dot{p}I_{xx} - \dot{q}I_{xy} - \dot{r}I_{xz} + qr\left(I_{zz} - I_{yy}\right) + \left(r^2 - q^2\right)I_{yz} - pqI_{xz} + rpI_{xy} \\
M_y &= -\dot{p}I_{xy} + \dot{q}I_{yy} - \dot{r}I_{yz} + rp\left(I_{xx} - I_{zz}\right) + \left(p^2 - r^2\right)I_{xz} - qrI_{xy} + pqI_{yz} \\
M_z &= -\dot{p}I_{xz} - \dot{q}I_{yz} + \dot{r}I_{zz} + pq\left(I_{yy} - I_{xx}\right) + \left(q^2 - p^2\right)I_{xy} - rpI_{yz} + qrI_{xz}
\end{aligned}
$$

Aerodynamic Coefficient Comparisons

The goal of the first phase of the F/A-18 aerodynamics database validation is to identify areas of the database that need improvement. With the vast amount of data collected for the validation, individual aerodynamic coefficient checks for each maneuver became time consuming and subjective. An acceptance criteria is being developed for such an analysis. The objective in developing the criteria is to create a standard definition for "good" and "bad" coefficient comparisons. Some studies have been conducted using a rating system based on the Theil coefficients, but have not yielded promising results. The study is on going.

Several preliminary analyses have been performed using the truth data set. Aerodynamic coefficients were extracted from the Navy's version 3.2 aerodynamics model using SCOPE. Figure 15 illustrates the coefficient comparison process using SCOPE.

One analysis investigated the aerodynamics model's fidelity at high roll rates during maneuvers at varying angles of attack. Figure 16 contains an example comparison of lateral and directional coefficients for a two-seat fighter escort high-roll-rate maneuver. The comparison reveals simulation over predictions of C_y and C_n as well as reasonable, but slight, under-predictions of Cl. Results such as these have lead to an in-depth study into the cause of the discrepancies. Results of the study suggested improvements to the model using dynamic modeling techniques[6].

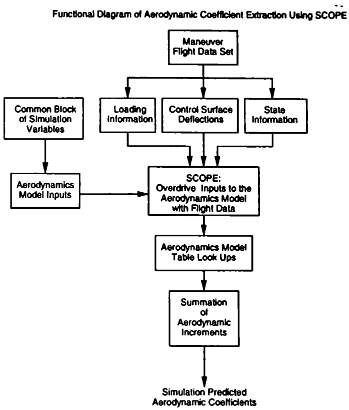

Functional Diagram of Aerodynamic Coefficient Extraction Using SCOPE
Maneuver Flight Data Set
Common Block of Simulation Variables
Loading Information
Control Surface Deflections
State Information
Aerodynamics Model Inputs
SCOPE: Overdrive Inputs to the Aerodynamics Model with Flight Data
Aerodynamics Model Table Look Ups
Summation of Aerodynamic Increments
Simulation Predicted Aerodynamic Coefficients

Fig. 15 Functional diagram of aerodynamic coefficient extraction from the aerodynamics model using SCOPE.

Summary and Conclusions

An aerodynamics model validation and baseline data set was created using data collected from two production F/A-18 aircraft. Particular care needed to be taken in the interpretation of the multiple production data sources available, for example the direct use of the sideslip angle as provided by the INS may have lead to misinterpretations during validation studies. The processes used to reconstruct and calibrate the data would have been facilitated by the addition of several sensors such as sideslip and a wide-range angle of attack probe. In general, the data are sufficient for validation purposes. Modifications to the angle-of-sideslip calibrations may be in order as investigations into a more elegant methods of bias determination continue.

The complete comprehensive validation of the F/A-18 aerodynamics model is on-going as is work to develop an acceptance criteria for aerodynamic coefficient comparisons.

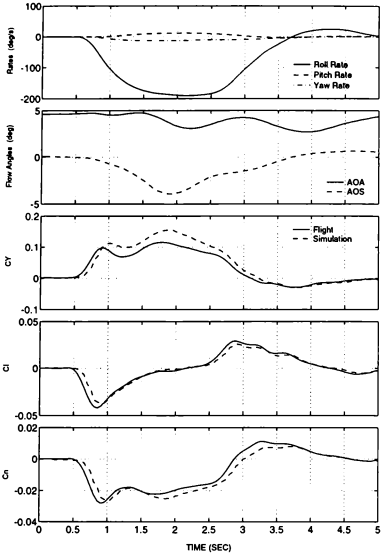


Fig. 16 Lateral-directional coefficient comparison of a two-seat fighter escort high-rate roll maneuver.

References

[1] Burton, R. A., Miller, C. C., and Mills, R E., "Manned Flight Simulator and the Impact on Navy Weapons Systems Acquisition," AIAA-94-3420-CP, August 1994.

[2] Fitzgerald, T.R., et al, "Improvements to the Naval Air Warfare Center Aircraft Division's F/A-18 Subsonic Aerodynamics Model", AIAA-94-3400 Flight Simulation Technologies Conference in Scottsdale, AZ, 1-3 August 1994.

[3] Hess, Robert A., "Subsonic F/A-18A and F/A-18B (TF-18A) Aerodynamics Identified from Flight Test Data", SCT 4522-220-1, July 1987.

[4] Hess, Robert A., "Development and Evaluation of a High-Fidelity F-18 Aerodynamic Model in a Unified Format"; SCT 6007-070-1, October 1989.

[5] O'Connor, Cornelius, "NATC F-18A/B Aerodynamic Math Model Modifications Incorporated during the Phase II Model Unification Effort"; BAR 90-13, September 1990.

[6] Ralston, J. N. et al, "Evaluation of the NAWCAD F/A-18 C/D Simulation Including Data Base Coverage and Dynamic Data Implementation Techniques," Bihrle Applied Research report number BAR 95-9, December 95.

[7] "NATOPS Flight Manual: Navy Model F/A-18A/B/C/D"; A1-F18AC-NFM-000, 15 January 1991; change 5 — 15 August 1993.

[8] Naval Air Warfare Center Aircraft Division Flight Test and Engineering Group Project Test Plan, " F/A-18 Simulation Validation and Verification Flight Test Program," August 1993.

[9] Klein, V., and Morgan, D., "Estimation of Bias Errors in Measured Airplane Responses Using Maximum Likelihood Method," NASA TM-89059, Jan. 1987.

[10] Pulley C., McDonnell Douglas Aerospace engineer, personal communication regarding informal internal McDonnell Aircraft Company memo titled "GPS Lever Arm Compensation," dated 3 Jan 1992, Fall 1994.

[11] Gingras, D. R., "Validation Documentation," SAIC Report No. 01-1393-3970-B003, March 1996.

[12] Maine, R.E, and Iliff, K.W., "Application of Parameter Estimation to Aircraft Stability and Control," NASA RP-1168, June 1986.

[13] Weiss, S. et al, "System Identification for X-31A Project Support — Lessons Learned so Far—", AGARD FMP Symposium "Flight Testing", Paper No. 14, Chania, Crete, Greece, 11-14 May 1992.

[14] Pepitone,D.D., Shively,R.J., and Bortolussi, M.R., "Pilot-Work-Load Prediction," Paper No. 871771, 6th SAE ABET Conference Long Beach, CA, 1988.

[15] Pindyck, R. S. and Rubinfield, D. L., *Econometric Models and Economic Forecasts Third Edition*, McGraw Hill, New Yaork, NY, 1991.

[16] West, G.R., "Actual Weight and Balance Report for Model F/A-18C Airplane Serial No.163985," MDC B1871, McDonnell Aircraft Company, St Louis, MO, Oct 1989.

AIAA-96-3530-CP

SIMULATION VALIDATION THROUGH LINEAR MODEL COMPARISON

Keith A. Balderson[*]
Donald P. Gaublomme[†]
Naval Air Warfare Center Aircraft Division
Code 432200A, MS-3
48140 Standley Road
Patuxent River, Maryland 20670

Justin W. Thomas[‡]
Science Applications International Corporation
Systems Technology Group
44417 Pecan Court, Suite B
California, Maryland 20619

Abstract

The Manned Flight Simulator at the Naval Air Warfare Center in Patuxent River, MD maintains high fidelity fixed and rotary wing simulation models. The aircraft simulations are utilized for a wide range of activities including flight test support, pilot training, and control law analysis and design. Validating aircraft math models against flight test data is an important part of the simulation process. Linear model comparison was used to validate the lateral-directional dynamic modes of the V-22 tilt-rotor aircraft in airplane mode. In this technique, linear model approximations of the simulation and aircraft dynamics are calculated independently and then compared. The simulation linear state-space model was extracted from the nonlinear V-22 simulation using a perturbation method. The aircraft linear state-space model was fit to flight test data from lateral-directional maneuvers using parameter identification tools. Time history comparisons were used to verify both linear models. Comparisons of the lateral-directional modes and the stability and control derivatives of the two models were made. The differences between the two models were used to locate potential problems with the nonlinear simulation.

Nomenclature

a_{1L}	Left rotor longitudinal flapping (deg)
a_{1R}	Right rotor longitudinal flapping (deg)
b_{1L}	Left rotor lateral flapping (deg)
b_{1R}	Right rotor lateral flapping (deg)
A	State-space system matrix
B	State-space control matirix
C	State-space output matrix
C_l	Nondimensional roll moment coefficient
C_N	Nondimensional yaw moment coefficient
C_Y	Nondimensional side force coefficient
D	State-space output control matrix
g	Gravity constant (ft/sec^2)
G	Cost function weighting matrix
I_{XX}	Moment of inertia about x body axes (slug-ft^2)
I_{YY}	Moment of inertia about y body axes (slug-ft^2)
I_{ZZ}	Moment of inertia about z body axes (slug-ft^2)
I_{XZ}	Product of inertia about x-z body axes (slug-ft^2)
J	Cost function matrix
L	Roll moment, body axes (ft-lb)
m	Aircraft mass (slug)
N	Yaw moment, body axes (ft-lb)
p	Roll rate perturbation, body axes (rad/sec)
r	Yaw rate perturbation, body axes (rad/sec)
t	Time (sec)
T	Total number of time steps
T_R	Roll mode time constant (sec)
T_S	Spiral mode time constant (sec)
u	State-space input vector
U	x velocity, body axes (ft/sec)
U_{Theil}	Theil coefficient
v	y velocity perturbation, body axes (ft/sec)
W	z velocity, body axes (ft/sec)
x	State-space state vector
y	State-space output vector
Y	Side force, body axes (lb)

[*] Aerospace Engineer
[†] Aerospace Engineer
[‡] Aerospace Engineer

Greek

β	Angle of sideslip (deg)
δ_a	Aileron position (deg)
δ_r	Rudder position (deg)
Δ	Denotes perturbed value
ζ	Damping ratio
ϕ	Roll attitude perturbation (rad)
Θ	Pitch attitude (rad)
ρ	Air density (slug/ft^3)
ω_n	Undamped natural frequency (rad/sec)

Subscripts

0	At reference flight condition

Superscripts

	Time rate of change
^	Estimated

Introduction

The V-22 tilt-rotor simulation at the Manned Flight Simulator (MFS) is a high fidelity nonlinear simulation utilized for flight test support, pilot training, and control law analysis. The airframe model is based on the Bell Helicopter Generic Tilt-Rotor simulation. The rotor model is a disk model with dynamic longitudinal and lateral flapping states. The rotor model structure is based on helicopter aerodynamic theory, flight test data, rotor test stand data, blade element model data, and airplane mode propeller efficiency data. The aerodynamic model is a component buildup of the aerodynamic subsystems including the fuselage, wing-pylon, horizontal tail, vertical tail, and landing gear. The aerodynamic model structure and data tables are based on wind tunnel data, theoretical equations, and flight test data.

The V-22 simulation is implemented in the MFS Controls Analysis and Simulation Test Loop Environment (CASTLE). CASTLE is a modular shell structure designed for simulation development, execution, and analysis. The modular design allows CASTLE to support a wide range of rotary and fixed wing aircraft. CASTLE provides a standard looping structure, equations of motion, and atmospheric models, as well as engineering analysis facilities for simulation validation.[1]

The data presented in this paper is part of an ongoing full envelope validation of the V-22 math model against flight test data. Initially, time history comparisons were used to validate the dynamic response of the V-22 simulation in airplane mode. For time history comparison, the simulation inputs are overdriven with flight test data, and the simulation outputs are plotted with measured flight test outputs. If the simulation outputs follow the flight test data within a desired tolerance then the simulation is validated. Overall the nonlinear simulation matched the airplane mode flight test data very well with one exception.

The simulation did not match the flight test data for lateral-directional maneuvers with the flaps set at twenty degrees. For this case, the time history plots alone did not provide enough information to understand the shortcomings of the simulation. The differences between the simulation model and flight test data for these lateral-directional maneuvers were investigated using linear model comparison.

Linear model comparison is a validation technique that provides dynamic mode and stability derivative information which can be used to update the full nonlinear simulation.[2] In this technique, linear model approximations of the simulation and aircraft dynamics are calculated independently and then compared. If the simulation model matches the flight test model within a desired tolerance then the simulation is validated.

A number of techniques are available to extract a linear model from a nonlinear simulation.[3] For this analysis, the simulation linear model was extracted using a perturbation technique. A second linear model representing the aircraft dynamics was fit to flight test data using time domain parameter identification (PID) tools.

Overview of the V-22 Osprey

The V-22 Osprey is a tilt-rotor aircraft capable of flight from hover to high speed airplane mode. Control of the V-22 is accomplished through both conventional airplane and helicopter controls, as illustrated by Figure 1. The airplane controls include an elevator, four flaperons, and two rudders. The helicopter controls include collective pitch, longitudinal cyclic, and lateral cyclic for each rotor.

The V-22 has a digital fly-by-wire control system consisting of a primary flight control system (PFCS) and an automatic flight control system (AFCS). The PFCS provides pilot control input shaping and essential feedback loops. The AFCS is designed to provide Level 1 handling qualities and auto-pilot functions.

In airplane mode, the aircraft is primarily controlled with the aerodynamic control surfaces; however, the

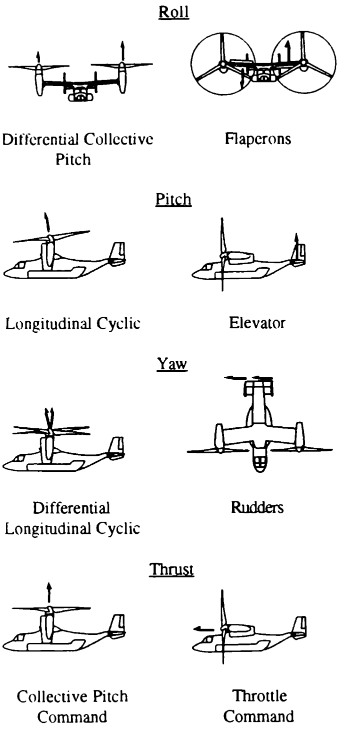


Figure 1. V-22 Controls: Helicopter (Left) and Airplane (Right).

rotor controls are still significant. The PFCS uses the rotor cyclic inputs to minimize the flapping angles of each rotor and the collective inputs to maintain a constant rotor speed and enhance thrust control.

State-Space Model Definition

The models developed in this study utilize a standard continuous time, state-space model.

$$\dot{\mathbf{x}} = A\mathbf{x} + B\mathbf{u} \tag{1}$$

$$\mathbf{y} = C\mathbf{x} + D\mathbf{u} \tag{2}$$

$$\mathbf{x}(t_0) = \mathbf{x}_0 \tag{3}$$

It is common practice to model the lateral-directional dynamics of an aircraft with a fourth order system.[4, 5] Typically in this type of model, the four states are the sideward velocity, roll rate, yaw rate, and roll angle. The inputs to the system are the rudder and aileron deflections.

To develop a linear model based on these states and inputs, the equations of motion for a rigid body are significantly simplified. The small perturbation approach is used to linearize the equations, and the higher order terms are neglected. It is assumed that the lateral-directional and longitudinal equations of motion may be separated.[4] Gyroscopic contributions from the rotors are neglected because the left and right rotors rotate in opposite directions. The lateral-directional dynamic equations are shown below.

$$Y_0 + \Delta Y + mg\phi\cos\Theta_0 = m(\dot{v} - W_0 p + U_0 r) \tag{4}$$

$$L_0 + \Delta L = I_{XX}\dot{p} - I_{XZ}\dot{r} \tag{5}$$

$$N_0 + \Delta N = -I_{XZ}\dot{p} + I_{ZZ}\dot{r} \tag{6}$$

$$\dot{\phi} = p + r\tan\Theta_0 \tag{7}$$

For unaccelerated initial conditions, the initial forces and moments (Y_0, L_0, and N_0) are zero. The perturbation force and moments can each be approximated with a first order Taylor expansion. The partial derivatives, Y., L., and N., are known as the dimensional stability and control derivatives.

$$\Delta Y = Y_v v + Y_p p + Y_r r + Y_{\delta r}\delta_r + Y_{\delta a}\delta_a \tag{8}$$

$$\Delta L = L_v v + L_p p + L_r r + L_{\delta r}\delta_r + L_{\delta a}\delta_a \tag{9}$$

$$\Delta N = N_v v + N_p p + N_r r + N_{\delta r}\delta_r + N_{\delta a}\delta_a \tag{10}$$

The cross-product of inertia, I_{XZ}, for the V-22 is generally less than 1% of the moments of inertia, I_{XX} and I_{ZZ}, so the roll-yaw coupling terms in equations 5 and 6 can be ignored without significantly affecting the model dynamics. This yields the final state equations for the rigid body dynamics.

$$\begin{aligned}\dot{v} = \frac{Y_v}{m}v + \left(\frac{Y_p}{m} - W_0\right)p + \left(\frac{Y_r}{m} + U_0\right)r + \\ g\cos\Theta_0\phi + \frac{Y_{\delta r}}{m}\delta_r + \frac{Y_{\delta a}}{m}\delta_a\end{aligned} \tag{11}$$

$$\dot{p} = \frac{1}{I_{XX}}\left(L_v v + L_p p + L_r r + L_{\delta r}\delta_r + L_{\delta a}\delta_a\right) \tag{12}$$

$$\dot{r} = \frac{1}{I_{ZZ}}\left(N_v v + N_p p + N_r r + N_{\delta r}\delta_r + N_{\delta a}\delta_a\right) \quad (13)$$

$$\dot{\phi} = p + r\tan\Theta_0 \quad (14)$$

For the V-22, additional states and inputs are required to model the dynamics of the rotor system. The states are the longitudinal and lateral flapping angles, and the inputs are the longitudinal and lateral cyclic pitch angles and the collective pitch angles.

Combining the rotor and rigid body dynamic models generates an 8th order state-space system. The A matrix can be divided into four quadrants as shown below, where the diagonal quadrants contain the pure rigid body and rotor dynamics, and the off diagonal quadrants represent the cross-coupling of the two systems.

$$A_{total} = \left[\begin{array}{c|c} A_{rigid\ body} & A_{rotor\ on\ rigid\ body} \\ \hline A_{rigid\ body\ on\ rotor} & A_{rotor} \end{array}\right] \quad (15)$$

The $A_{rigid\ body}$ quadrant is defined by equations 11-14.

Similarly, the control matrix, B, can be separated into quadrants for the control surface and rotor input effects on the rigid body and rotor states.

$$B_{total} = \left[\begin{array}{c|c} B_{surfaces\ on\ rigid\ body} & B_{rotor\ inputs\ on\ rigid\ body} \\ \hline B_{surfaces\ on\ rotor} & B_{rotor\ inputs\ on\ rotor} \end{array}\right] \quad (16)$$

The upper left quadrant is defined by equations 11-14. Since the rigid body control surfaces have no effect on the rotor flapping, the elements of the lower left quadrant are zeros. The cyclic inputs have very little effect on the rigid body motion, but the collective inputs for each rotor have a significant effect on the yaw rate.

The C matrix is nearly identity since the output vector is approximately equivalent to the state vector. The matrix does contain terms to transform the flapping angles to the axes used in flight test measurements. The D matrix is identically zero since the inputs do not directly affect the outputs.

Linear Model Extraction from the Nonlinear Simulation

The V-22 simulation in the CASTLE architecture was initialized at the flight test initial conditions. The initial conditions were level steady-state flight, nacelles fixed in airplane mode, true velocity of 183 knots, and flaps set at twenty degrees. A linear model was obtained from the full nonlinear simulation using the Linear Model Extraction (LME) facility.[3]

LME uses the offset derivative method to extract a linear model. In this method, perturbations are added to each input and state, and the resulting changes in outputs and state derivatives are recorded. Integration is frozen within the simulation so that the state derivatives are not allowed to propagate. The state-space matrices are then computed.

$$A_{ij} = \frac{\Delta\dot{x}_i}{\Delta x_j} \qquad B_{ik} = \frac{\Delta\dot{x}_i}{\Delta u_k}$$
$$\qquad (17)$$
$$C_{nj} = \frac{\Delta y_n}{\Delta x_j} \qquad D_{nk} = \frac{\Delta y_n}{\Delta u_k}$$

In the above equation, i is the state derivative index, j the state index, k the control input index, and n the output index.

Since the simulation in general is nonlinear, the change in outputs and state derivatives for different size perturbations may not be perfectly linear. The user specifies four perturbation sizes for each input and state. LME perturbs the simulation positively and negatively for each of the perturbation sizes and picks the perturbation size that results in the best linearity. A typical example of an LME result for a linear relationship is shown in Figure 2.

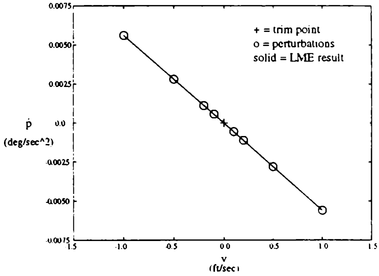


Figure 2. LME result for a linear relationship between the state derivative and perturbation parameter.

In some cases the LME result shows that a linear relationship between an output and a perturbation does not exist. An example of a nonlinear relationship is shown in Figure 3. This figure shows that the linear approximation of the nonlinear data is adequate near the reference condition. The linearity information is useful in determining the quality of the linear model with respect to each coefficient. The output of LME is the state-space model and the linearity information for each coefficient.

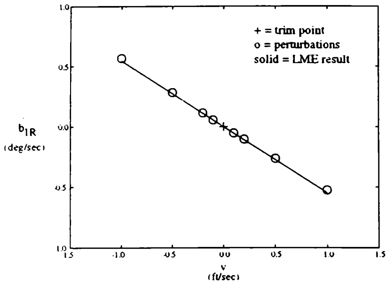


Figure 3. LME result for a nonlinear relationship between the state derivative and perturbation parameter.

Verification of the LME Model

The state-space LME model was compared to the full nonlinear simulation to examine the quality of the extracted linear model. Two input maneuvers, a lateral stick ramp and a rudder doublet, were used to verify the linear model. Each model was driven with the same inputs, and the model outputs were compared. The control inputs and the model outputs are shown in Figure 4 as a concatenated data set. The figure shows that the linear model accurately represents the lateral-directional dynamics of the full nonlinear simulation about the reference flight condition for the two maneuvers.

Linear Model Identification to Flight Test Data

Three lateral-directional flight test maneuvers, a roll reversal, a yaw reversal, and a yaw doublet, were used for the parameter identification. Visual inspection of the 30 degree roll reversal showed good excitation of the roll mode and yaw due to roll characteristics. The yaw reversal and yaw doublet showed good excitation of the dutch roll mode on the sideward velocity and yaw rate time histories. The flight test data was preprocessed and analyzed for kinematic consistency. The final data set showed good consistency with the rigid body kinematic equations. The three maneuvers were concatenated for parameter identification.

Parameter identification was limited to stability and control derivatives from the rigid body lateral-directional equations contained in the upper left hand quadrants of the A and B matrices. No parameters in the rotor dynamic equations or the cross-coupling matrices were included in the identification. Identification of rotor dynamic parameters from maneuvers with mainly conventional airplane control inputs was not attempted. The LME state-space matrices were used as the initial parameter values in the identification.

The parameters available for identification formed a set of nine stability derivatives and six control derivatives. Cramer-Rao bounds, which are estimates of the standard deviation of the identified parameters, were used to assess the identifiability of each parameter.[6] Three stability derivatives and one control derivative had relatively high Cramer-Rao bounds and were not identified. These parameters were held constant at the LME model value.

The cost function was defined as a weighted sum of squared errors between the measured outputs, $\mathbf{y}$, and the estimated outputs, $\hat{\mathbf{y}}$. The weighting matrix, G, is a diagonal matrix that contains the relative weighting for each output.

$$J = \sum_{i=1}^{T} \left[G(\mathbf{y}_i - \hat{\mathbf{y}}_i)^2 \right] \tag{18}$$

The rigid body states were weighted ten times greater than the rotor flapping states for the identification. The rotor flapping outputs were included in the cost function because the rigid body equations and rotor dynamic equations are coupled.

The cost function was minimized through parameter identification using the MATLAB® computing environment. The state-space model was overdriven with flight test inputs to produce the estimated outputs. A Levenburg-Marquardt routine, which is a combination of the Gauss-Newton and steepest descent methods, was used to minimize the cost function.[7]

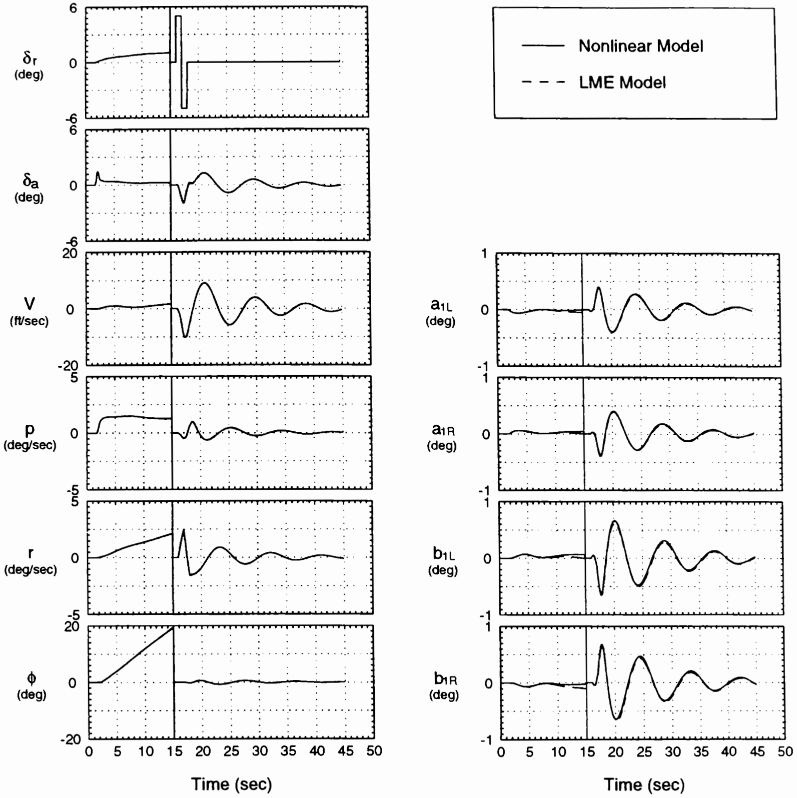


Figure 4. Verification of the LME model to the full nonlinear simulation.

The measured inputs and outputs from flight test, the LME model outputs, and the PID model outputs are shown in Figure 5. A marked improvement in the identified model is evident from the plots of the outputs.

The improvement of the identification was quantitatively assessed using goodness of fit statistics applied to the measured and modeled outputs. Theil's inequality coefficient is a statistic that scales the root mean square error between zero and one. An inequality coefficient value of zero indicates a perfect fit, and a value of one indicates the fit is as bad as possible.[8]

$$U_{Theil} = \frac{\sqrt{\frac{1}{T}\sum_{i=1}^{T}(\mathbf{y}_i - \hat{\mathbf{y}}_i)^2}}{\sqrt{\frac{1}{T}\sum_{i=1}^{T}(\mathbf{y}_i)^2} + \sqrt{\frac{1}{T}\sum_{i=1}^{T}(\hat{\mathbf{y}}_i)^2}} \qquad (19)$$

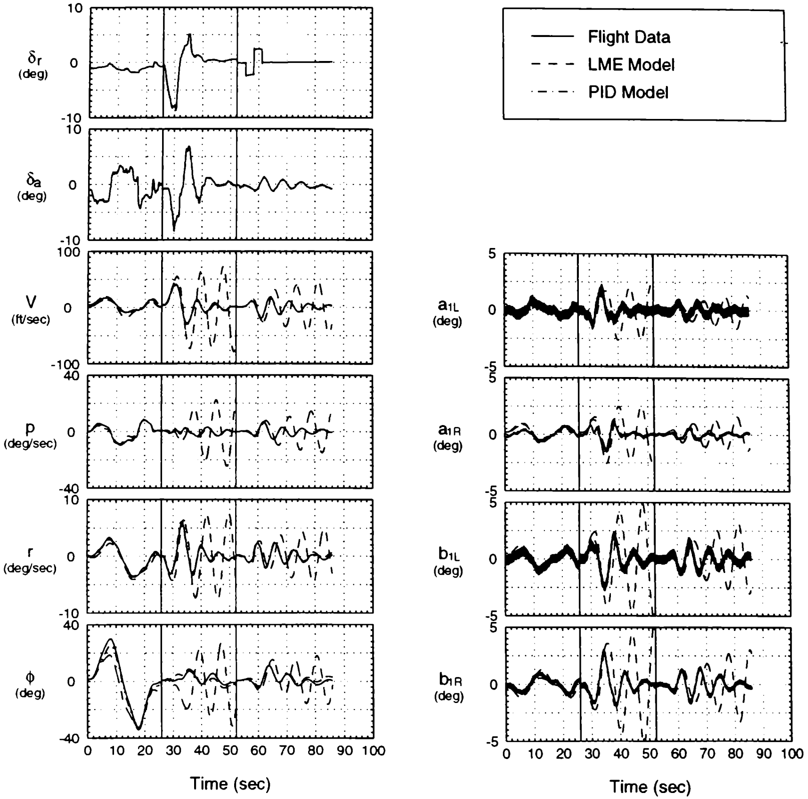


Figure 5. Flight test data and LME and PID model outputs for the parameter identification maneuvers.

Goodness of fit statistics showing the improvement from the initial model to the identified model are presented in the first part of Table 1.

Verification of the PID Model

The PID state-space model was verified against independent flight test data to assess the quality of the model. Often a model will match the data used in the identification but will not accurately predict independent data. Good prediction capability provides confidence that the identified model represents the actual aircraft dynamics.

A yaw doublet not included in the identification was used to verify the identified model. The identified model was overdriven with the inputs from the yaw doublet and the outputs were compared to the flight test data. Figure 6 shows the flight test and model outputs.

Table 1. Goodness of fit statistics comparing the LME outputs and the PID model outputs to flight test data used in the identification and independent verification data.

State Space Model Outputs	U_{Theil} LME Model Identification Data Set	U_{Theil} PID Model Identification Data Set	U_{Theil} PID Model Verification Data Set
v	0.679	0.079	0.210
p	0.742	0.072	0.256
r	0.586	0.079	0.227
ϕ	0.491	0.094	0.293
a_{1R}	0.674	0.388	0.411
a_{1L}	0.685	0.310	0.345
b_{1L}	0.728	0.223	0.217
b_{1R}	0.723	0.145	0.255

The PID model adequately represents the verification data at the beginning of the maneuver while the rudder doublet is active. The model is somewhat overdamped during the remainder of the maneuver. Goodness of fit statistics for the verification yaw doublet are presented in the last column of Table 1. As expected, the Theil coefficients for the verification data set are higher than those for the identification data set, but they still show an improvement over the initial LME model.

The identification process produced a state-space model with adequate dutch roll prediction capability. Less confidence was placed in the roll mode predictability for two reasons. The thirty degree roll reversal used in the identification stretches the linearity assumption, and independent flight test data was not available to verify the roll response.

Comparison of Linear Models

An obvious conclusion that could be drawn from Figure 5 is that the LME state-space model does not accurately represent the dutch roll mode of the aircraft. The time history plots for the yaw reversal from the LME model suggest a slightly unstable dutch roll mode. Quantifying this problem using only a time history comparison would be very difficult.

Table 2. Lateral-directional dynamic stability characteristics for the LME and PID models.

	Dutch Roll Mode		Spiral Mode	Roll Mode
	ζ	ω_N	T_S	T_R
LME Model	-0.004	0.820	7.94	1.26
PID Model	0.041	1.005	8.87	1.37

Table 3. Nondimensional stability derivative values from the LME and PID models.

Stability Derivative	LME Model	PID Model	Difference
$C_{Y\beta}$	-2.41	-2.42	-0.01
C_{Yp}	-0.464	-0.464	No ID
C_{Yr}	0.506	0.506	No ID
$C_{Y\delta r}$	-0.236	-0.240	-0.004
$C_{Y\delta a}$	-0.226	-0.226	No ID
$C_{l\beta}$	-0.342	-0.406	-0.064
C_{lp}	-1.20	-1.18	0.02
C_{lr}	-0.169	-0.169	No ID
$C_{l\delta r}$	-0.017	-0.072	-0.055
$C_{l\delta a}$	-0.350	-0.320	0.030
$C_{N\beta}$	0.042	0.138	0.096
C_{Np}	-0.640	-0.683	-0.043
C_{Nr}	-1.77	-1.85	-0.08
$C_{N\delta r}$	0.173	0.159	-0.014
$C_{N\delta a}$	-0.0217	-0.0400	-0.0183

An inherent strength of the linear model comparison method is the ability to compare the dynamic modes contained in the state-space models. The eigenvalues of the A matrix which describe the modes of the aircraft are easily calculated in MATLAB®. The eigenvalues of the lateral-directional rigid body equations are shown on a complex plane plot in Figure 7. The damping ratio and undamped natural frequency of the dutch roll mode, along with the time constants of the spiral and roll modes, are shown in Table 2. The acceptable tolerance

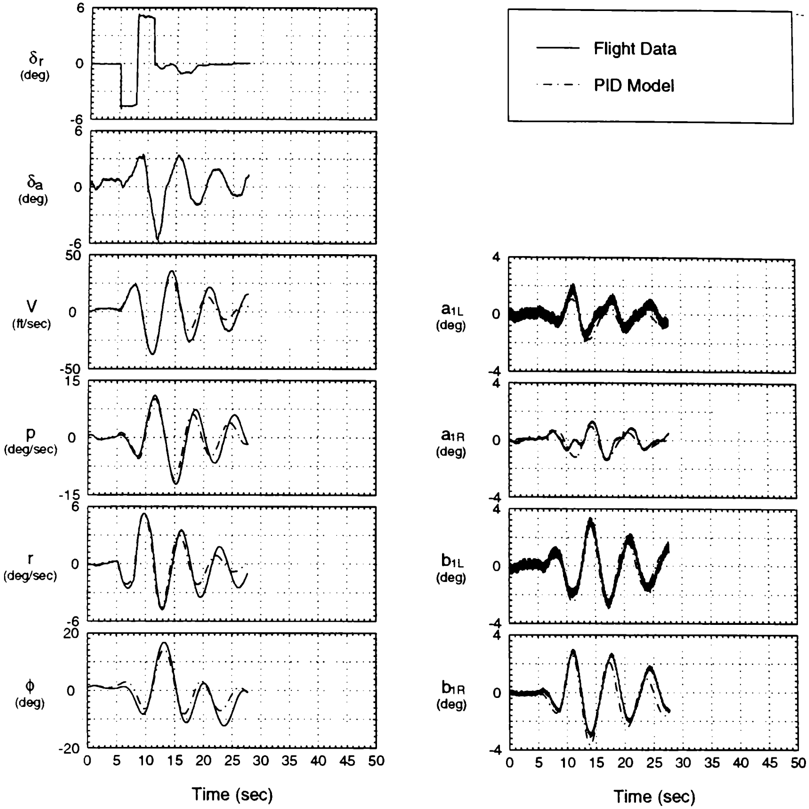


Figure 6. Flight test data and PID model outputs for an independent verification yaw doublet.

for error in the mode comparison depends on the specific aircraft and simulation application. A general tolerance of 5% has been proposed for this type of analysis.[2]

Both the LME and PID models contain classic second order dutch roll modes. The dutch roll mode of the LME model is slightly unstable as evident from the negative damping ratio. The dutch roll damping ratio of the identified model is larger in magnitude and positive. The LME dutch roll natural frequency is 18% less than the dutch roll natural frequency of the PID model. Based on the proposed tolerance, the LME model is not a valid representation of the dutch roll mode exhibited in flight test.

The LME and PID models both contain first order spiral and roll modes. The spiral mode time constant of the LME model is 10% less than that of the PID model. The roll mode time constant of the LME model is 8% less than the PID model value.

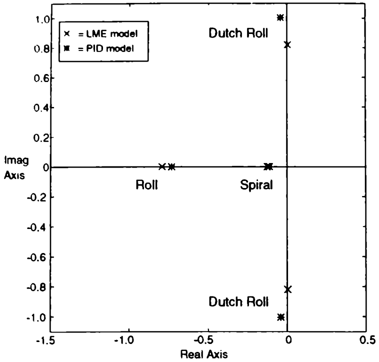


Figure 7. Complex plane plot of the lateral-directional dynamic modes from the LME and the PID models.

In addition to the dynamic mode information, the aircraft stability and control derivatives can be calculated from the state-space models. The dimensional stability derivatives are imbedded in the elements of the state-space model as shown in equations 11-13. The rigid body stability and control derivatives were extracted from the A and B matrices and nondimensionalized. Table 3 contains the nondimensional stability and control derivatives calculated from the LME and identified models.

Simulation Stability Derivative Investigation

Two approaches can be taken to incorporate the increments in stability and control derivatives into the nonlinear simulation. The most straightforward approach is to directly add the increments into the aerodynamic force and moment equations. This approach is only appropriate for simple simulation models. The nonlinear nature of the V-22 simulation does not lend itself to this direct method. For complex models such as the V-22, the stability and control derivative information can be used to locate weaknesses in the physically based equations of the simulation model.

The shaded stability derivative in Table 3, $C_{N\beta}$, showed the greatest difference between the LME and PID models. $C_{N\beta}$ is the nondimensional yawing moment coefficient due to sideslip. $C_{N\beta}$ must be positive for the aircraft to exhibit directional stability. In the nonlinear simulation, this coefficient contains contributions from the aerodynamic and rotor models. The signs and magnitudes of these contributions can be examined in the LME environment.

Examination of the $C_{N\beta}$ components showed that the rotor model contribution is destabilizing (negative) and the aerodynamic model contribution is stabilizing (positive). The rotor model contribution is only slightly smaller in magnitude than the aerodynamic model contribution. The rotor and aerodynamic models combined to produce a total simulation $C_{N\beta}$ that provides weak directional stability.

Investigation into the rotor model equations showed that the magnitude of the rotor model $C_{N\beta}$ contribution is dependent on a sideward velocity effect model. This sideward velocity effect is intended to model the interference effect that one rotor would have on the other. For high speed airplane mode flight, physical intuition suggests that the interference effect between the rotors should be negligible. Elimination of the sideward velocity effect in the nonlinear simulation decreased the rotor destabilizing effect on the directional stability, and the total $C_{N\beta}$ moved closer to the PID predicted value.

Further investigation is required to determine the history of the sideward velocity effect and assess the validity of the effect for airplane mode flight. In this investigation the linear model comparison technique has provided valuable insight into the validation process and may have located a weakness in the rotor model physical equations.

Conclusions

A linear model was extracted from the V-22 full nonlinear simulation using LME. This model was verified against time history data from the nonlinear simulation. The linear model accurately represented the lateral-directional dynamics of the verification maneuvers. A second linear model was fit to flight test data using parameter identification tools. This model was verified using an independent maneuver not included in the identification process. The linear model exhibited adequate predictability of the dutch roll mode dynamics.

The lateral-directional dynamic modes of the two linear models were compared. This comparison illustrated a significant difference between the dutch roll modes for the LME and PID models. The stability derivatives for the two models were calculated and compared. The $C_{N\beta}$ derivative was most significantly changed between the two models.

The linear model comparison yielded dynamic mode and stability derivative information unavailable from a standard time history comparison. This information was used to investigate a possible error in the nonlinear simulation math model. Within the scope of this paper, the linear model comparison technique proved to be a valuable tool for validating a complex nonlinear simulation.

Recommendations

The flight test data used in this analysis originated from V-22 envelope expansion flights. The linear model comparison technique could be used more effectively with flight test data better suited to parameter estimation. The control inputs should be excited independently, and maneuvers with small perturbations about the reference condition should be conducted. Multiple maneuvers should be flown to provide sufficient data for identification and verification.

References

[1] Nichols, J. H., "Controls Analysis and Simulation Test Loop Environment (CASTLE) User's Guide CASTLE Version 3.0." Naval Air Warfare Center Aircraft Division, Patuxent River, MD, 1993.

[2] Padfield, G.D., "Simulation Model Validation," AGARD AR-280, 1991, pp. 223-234.

[3] Balderson, Keith and Weathers, Jeffrey, "Comparison of Frequency Response and Perturbation Methods to Extract Linear Models from a Nonlinear Simulation," AIAA Flight Simulation Technologies Conference, 1994.

[4] Etkin, Bernard, "Dynamics of Flight - Stability and Control," John Wiley and Sons, New York, 1982.

[5] McLean, Donald, "Automatic Flight Control Systems," Prentice Hall, New York, 1990.

[6] Maine, Richard E., Iliff, Kenneth W., "Application of Parameter Estimation to Aircraft Stability and Control - The Output Error Approach," NASA RP-1168, 1986.

[7] Grace, Andrew, "Optimization Toolbox User's Guide," The Math Works Inc., 1992.

[8] Pindyck, Robert S., and Rubinfeld, Daniel L., "Econometric Models and Economic Forecasts," McGraw-Hill, Inc., New York, 1991.

AIAA-96-3531-CP

GN&C INTEGRATION AND TEST FACILITY: MITIGATING RISK IN ISS GN&C SYSTEM DEVELOPMENT

D. Petri, S. Sepahban, K. Frank, H. Hu

National Aeronautics & Space Administration
Johnson Space Center
Houston, TX 77058

Abstract

The International Space Station (ISS) Guidance, Navigation and Control system functionality is distributed across the US Flight Segment and the Russian Flight Segment. This includes the distribution of sensors, effectors, computers and application software that must work together to operate as a fully capable system. Given the complicated and critical nature of the GN&C system, the ISS Program Office and the Johnson Space Center (JSC) agreed that a test-bed facility would be developed and early integration and testing of the end-to-end ISS GN&C system would be performed to mitigate the risks associated with the ISS GN&C integration and test. *This facility, the GN&C Integration and Test Facility (GITF), is the only facility in the world capable of functional integration and end-to-end testing of the US and Russian on-orbit GN&C hardware and software, including their interfaces with the C&C system.* This paper elaborates on the testing requirements, the test facility architecture, status-to-date and presents insights into how the facility can be used beyond the GN&C development phase.

Introduction

The International Space Station (ISS) Guidance, Navigation and Control (GN&C) system functionality is distributed across the US Flight Segment and the Russian Flight Segment. This includes the distribution of sensors, effectors, computers and application software that must work together to operate as a fully capable system. Similarly, the Command and Control system is distributed across both flight segments. In addition, these systems are being built by multiple contractors, assembled in stages on-orbit and will be commanded by ground controllers and on-board crew members. The combination of these factors complicate the processes for system integration and verification of the International Space Station GN&C system.

Given the complicated and critical nature of the GN&C system, the ISS Program Office and the Johnson Space Center (JSC) Engineering Directorate agreed that the Aeroscience and Flight Mechanics Division (A&FMD) would lead the development of a test-bed facility and co-sponsor early integration and testing of the end-to-end ISS GN&C system to mitigate the risks associated with the ISS GN&C integrated test and verification plan, including:

- Closed-loop, end-to-end US GN&C testing , including the Global Positioning System (GPS) and the Rate Gyro Assembly (RGA) in the loop
- Testing of the Control Moment Gyro (CMG)/GN&C software interaction
- Testing of US GN&C and Command and Control (C&C) interfaces prior to joint testing in Russia (scheduled for Fall 1996)
- Time limitations and facility loading of the Software Verification Facility (SVF) for GN&C testing (Note that this facility does not use any GN&C sensor or effector hardware: only application software, and sensor and effector simulation models)
- Complete functional testing of US and Russian Segment (RS) GN&C prior to formal verification activities at the SVF.

This facility, the **GN&C Integration and Test Facility (GITF)**, is the only facility in the world capable of functional integration and end-to-end testing of the US and Russian on-orbit GN&C hardware and software, including their interfaces with the C&C system.

Facility Description

The current GITF architecture consists of a mix of functional equivalent units (FEU) of flight hardware, hardware emulations and simulation models, GN&C and C&C flight software, specialized test equipment, and computer workstations provided by ISS Program elements, and the A&FMD and Avionics Divisions of the JSC Engineering Directorate of NASA/JSC. These elements are connected together by appropriate 1553, 488, 422, analog, discrete and Ethernet data bus connections (see Figure 1). The planning schedule of equipment availability is shown in Figure 2. Specifically, this equipment consists of:

US GN&C FEU Multiplexer/Demultiplexer (MDM) – This computer runs the US GN&C flight software and interfaces to the US GN&C sensor and effectors, to the US C&C MDM, and to the Russian GN&C Fault Containment Region (FCR). (Provided by Honeywell)

Current Status: The GN&C Extended MDM has been installed and accepted into the GITF, as of June 1, 1996.

US C&C FEU MDM – This computer runs the US C&C flight software and interfaces with all US systems and communicates with the Russian C&C system. (Provided by Honeywell)

Current Status: The C&C MDM is expected to be delivered to the GITF by July 1, 1996.

Russian GN&C FEU FCR – This computer runs the RS GN&C flight software and interfaces to the RS GN&C sensors and effectors, the RS C&C FCR and to the US GN&C MDM. (Provided by ESA)

Current Status: Negotiations are proceeding with the ISS Program Office, the JSC Engineering Directorate and Daimler-Benz (designer and manufacturer) for the

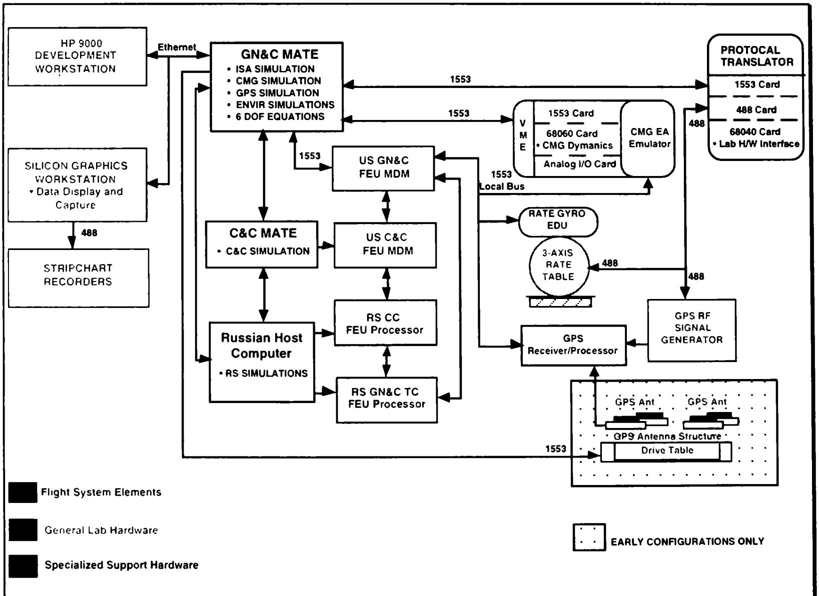


Figure 1: GN&C Integration & Test Facility Architecture

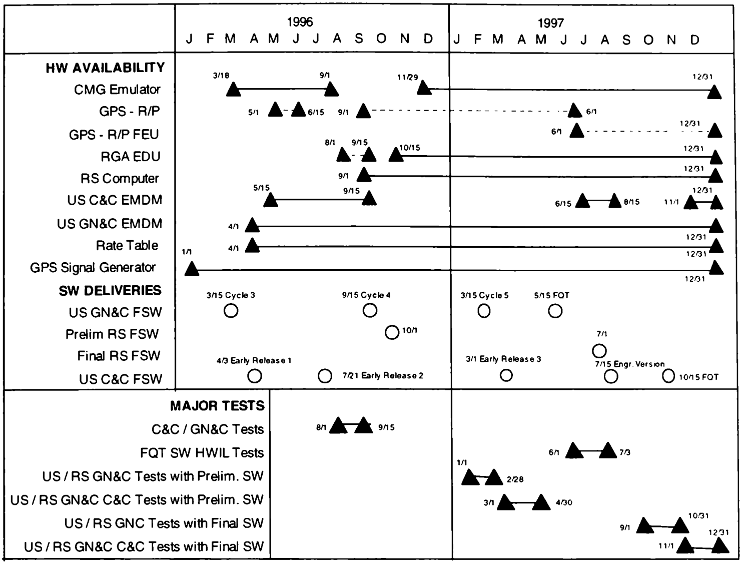


Figure 2: GN&C Integration & Test Facility Planning Schedule

procurement of the computer, as well as the application and test support equipment.

Russian C&C FEU FCR – This computer runs the RS C&C flight software and interfaces with all RS systems, as well as communicates with the US C&C system. (Provided by ESA)

Current Status: Negotiations are proceeding with the ISS Program Office, the JSC Engineering Directorate and Daimler-Benz (designer and manufacturer) for the procurement of the computer, as well as the application and test support equipment.

MDM Application and Test Equipment (MATE) – This workstation provides the tools for development of the ISS flight software (including the use of code development and autocoding capabilities using the Integrated Systems Incorporated MATRIXx tool set). It performs the cross-compilation and download of the flight software to the MDM. It also supports real-time simulations using environment models and the MDM. The verified system models will be resident on this workstation. (Provided by Honeywell)

Current Status: The MATE has been installed and accepted into the GITF, as of May 10, 1996.

CMG Emulator – This VME chassis contains a single-string duplication of the CMG electronics assembly and a simulation model of the CMG mechanical assembly. (Developed in-house by NASA/JSC Engineering Directorate)

Current Status: The CMG dynamics simulation model has been completed as is ready to be integrated with the electronics assembly. All boards of the electronics assembly have been fabricated and accepted. Integration, checkout and acceptance of the simulation model and the electronics assembly is expected to be complete by July 1, 1996.

RGA FEU – This is a functional equivalent unit of the ISS Rate Gyro Assembly. (Provided by Honeywell)

Current Status: The RGA FEU is expected to be delivered to the GITF by August 1, 1996.

3-Axis Rate Table – The RGA will be mounted to the rate table and the rate table will be commanded through the environment simulation to stimulate the RGA sensor package. (Provided by NASA/JSC Engineering Directorate)

Current Status: The 3-Axis Rate Table was delivered to the GITF on May 22, 1996. Installation, alignment and final acceptance is expected to be complete by June 7, 1996.

GPS Receiver/Processor (GPS R/P) – The GPS receiver/processor is a government furnished piece of equipment (GFE) to be used for position and attitude determination on-board the ISS. (Developed in-house by NASA/JSC Engineering Directorate)

Current Status: The GPS R/P is in development for use on the ISS. A prototype system, available from the JSC Avionics Division, is currently being used to emulate GPS R/P capabilities in the GITF.

GPS Signal Generator (GPS S/G) – The GPS Signal Generator models the GPS satellite constellation and ISS orbital dynamics. It then generates signals that duplicate the RF signatures received by the GPS antennas to be processed by the GPS Receiver/Processor. (Provided by NASA/JSC Engineering Directorate)

Current Status: The GPS S/G was delivered to the GITF on January 15. After extensive training on its use, it has been used for testing the GPS R/P for obtaining state information, including vehicle attitude determination. Most recently.

GPS Rooftop Fixture (GPS R/F) – The GPS Rooftop Fixture consists of 4 GPS antennas mounted on a positioner. This configuration can be slewed to simulate relative motion while accepting actual GPS satellite data. It is used to provide calibration data and comparison data from the GPS S/G. (Provided by NASA/JSC Engineering Directorate)

Current Status: The GPS R/F has been fabricated by the Manufacturing Division and has been delivered and installed on the roof of Building 16 at JSC.

Protocol Translator – This VME chassis contains the computers and software for translating various data bus protocols in order to properly interface the MATE, CMG Emulator, RGA, Rate Table, GPS signal Generator and GPS Rooftop Fixture. (Developed in-house by NASA/JSC Engineering Directorate)

Current Status: The Protocol Translator is currently in development and is expected to be delivered by July 1, 1996.

Data Capture & Display - In addition to the data capture and display capabilities of the MATE, the GITF provides additional capabilities through the use of strip chart recorders and a SGI workstation. These devices provide electronic data capture, parameter displays over time in both paper and electronic formats, and 3-D solid model graphical representations of ISS showing position, attitude and other vehicle state information.

Current Status: The SGI and strip chart recorders have been delivered to the GITF. Data connectivity has been established with the MATE workstation. Data connectivity from the MATE real-time computer is expected to be complete by 1 July 1996.

In addition to these hardware components, there is an extensive set of software elements (the functional interfaces are summarized in Figure 3):

US GN&C Flight Software - This is the GN&C software providing the following functions:

- GN&C Application Software
  - Navigation
  - Attitude Determination
  - Pointing and Support
  - Attitude Control
  - Command & Moding
- Common Utilities Software
  - Fault Detection, Isolation & Recovery
  - Common MDM Hardware Utilities

(Provided by MDAC)

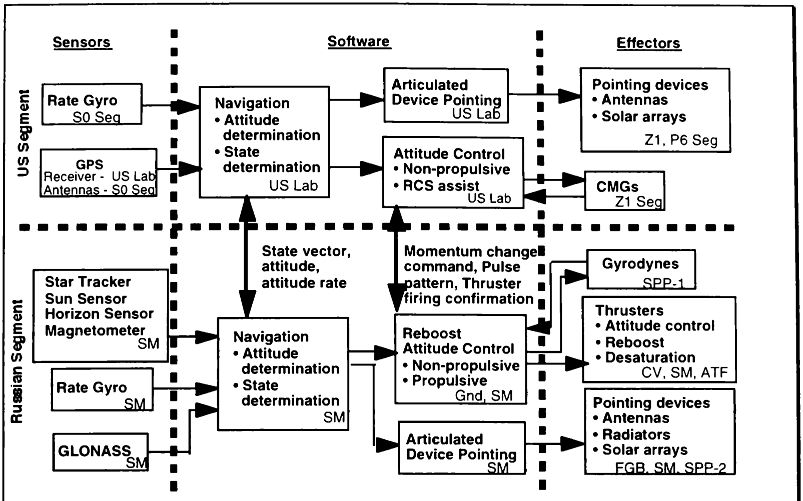


Figure 3: International Space Station GN&C Functional Interfaces

Current Status: The Cycle 3 version of the flight software has been delivered to GITF. Software checkout and initial hardware integration is in process.

US GN&C FEU MDM Operating System - This is the real-time operating system for the GN&C MDM. (Provided by Honeywell as part of the GN&C MDM)

Current Status: The GN&C Extended MDM Operating System was delivered with the MDM, which has been installed and accepted into the GITF, as of June 1, 1996.

US GN&C System Models - These are math models of the US GN&C hardware elements, GPS, RGA and CMG. These math models are used when the GITF hardware is unavailable. (Provided by MDAC as part of their software deliveries. Enhanced models provided by JSC)

Current Status: The GN&C System Models were delivered as part of the Cycle 3 software delivery to GITF.

Simulation Environment and Vehicle Models - These are math models of the space environment and ISS vehicle characteristics needed for simulation, including:

- Six Degrees of Freedom Equations of Motion
- Kinematic Calculations
- Gravity and Gravity Gradient Torque Models
- Atmospheric Density Model
- Aerodynamics Model
- Solar Ephemeris Model
- TDRSS Ephemeris Model
- Earth Blockage Model
- Payload Models
- Time Calculation

(Provided by MDAC as part of their software deliveries. Enhanced models provided by JSC)

Current Status: The GN&C Environment and Vehicle Models were delivered as part of the Cycle 3 software delivery to GITF.

US C&C Flight Software - This is the C&C application software providing all command and control services between US MDMs and Russian FCRs. (Provided by Boeing)

Current Status: Version 1.0 of the C&C Flight Software is expected to be delivered to the GITF by August 1, 1996.

US C&C FEU MDM Operating System - This is the real-time operating system for the C&C MDM. (Provided by Honeywell as part of the C&C MDM)

Current Status: The C&C MDM Operating System is expected to be delivered with the C&C MDM to the GITF by July 1, 1996.

US C&C System Models - These are the math models of other US systems that are required to make the C&C software functional. (Provided by Boeing as part of their software deliveries. Enhanced models provided by JSC and the ISS Program Office)

Current Status: C&C System Models are expected to be delivered with the C&C Flight Software to the GITF by August 1, 1996.

Russian GN&C Flight Software - This is the Russian GN&C application software. This includes:

- Reboost Logic
- Desaturation Logic
- Sequence Logic
- Attitude Controller
- Attitude Determination
- Input/Output Services (1553 boxcar interface)

(Provided by RSA. Preliminary functional Russian flight software provided by MDAC. Enhanced functional Russian flight software provided by JSC)

Current Status: Negotiations are proceeding with the ISS Program Office and the Russian Space Agency for delivery to GITF. Preliminary functional flight software is provided as part of the MDAC Cycle 3 US software delivery.

Russian GN&C FEU FCR Operating System - This is the real-time operating system for the GN&C FCR. (Provided by ESA as part of the GN&C FCR)

Current Status: Negotiations are proceeding with the ISS Program Office, the JSC Engineering Directorate and Daimler-Benz (designer and manufacturer) for delivery of the GN&C FCR Operating System to be delivered as part of the procurement of the computer.

Russian GN&C System Models - These are math models of the Russian GN&C sensor and effector hardware. This includes:

- Payload Motion (Russian RMS)
- Thrusters
- Gyrodynes

(Provided by RSA as part of their software deliveries. Preliminary models provided by MDAC. Enhanced models provided by JSC)

Current Status: Negotiations are proceeding with the ISS Program Office and the Russian Space Agency for delivery to GITF. Preliminary models are provided as part of the MDAC Cycle 3 US software delivery. Enhanced system models are being developed by JSC as an augmentation to existing US/RS GN&C analysis tasks.

Russian C&C Flight Software - This is the C&C application software providing all command and control services between US MDMs and Russian FCRs. (Provided by RSA)

Current Status: Negotiations are proceeding with the ISS Program Office and the Russian Space Agency for delivery to GITF. Preliminary functional flight software is provided as part of the MDAC Cycle 3 US software delivery.

Russian C&C FEU FCR Operating System – This is the real-time operating system for the C&C FCR. (Provided by ESA as part of the C&C FCR)

Current Status: Negotiations are proceeding with the ISS Program Office, the JSC Engineering Directorate and Daimler-Benz (designer and manufacturer) for delivery of the C&C FCR Operating System to be delivered as part of the procurement of the computer.

Russian C&C System Models - These are the math models of other Russian systems that are required to make the C&C software functional. (Provided by RSA as part of their software deliveries)

Current Status: Negotiations are proceeding with the ISS Program Office and the Russian Space Agency for delivery to GITF. Preliminary models are provided as part of the MDAC Cycle 3 US software delivery.

Integrated Testing

The testing to be accomplished in the GITF is broken into four major pieces:

1. Site Acceptance
2. US GN&C System Testing
3. US GN&C and C&C System Testing
4. US/RS GN&C and C&C System Testing

The first set deals with site acceptance testing. The purpose of the site acceptance tests will be to ensure the GITF is ready to conduct the development testing. These acceptance test will be run every time a new hardware element is added to the GITF configuration or after a new software update is incorporated. The site acceptance test results will be compared to the baseline results obtained from GN&C analysis or test data synthesized from other test results.

The second set deals with the US GN&C system and the interfaces between the US GN&C Processor/Application, US GN&C hardware elements, US GN&C MDM, US C&C MDM. Specifically, this includes:

- US Flight Software
- GPS integration testing
- RGA integration testing
- CMG integration testing
- Integrated USOS GN&C testing

These tests will ensure the proper data is transferred in the proper format and at the proper frequencies, such as command and data content and timing across the interfaces. The ORU hardware or certified math models may be used based upon resource and hardware availability.

GN&C Processor/Application (GN&C P/A) requirements testing will consist of compliance testing of specific GN&C P/A requirements to ensure the US GN&C P/A performs as per the requirement specifications.

The third set is integrated testing of the US GN&C and US C&C systems and their common interfaces, and includes:

- USOS GN&C Moding
- USOS Fault Detection, Isolation and Recovery (FDIR) testing
- GN&C Uplink/Downlink testing
- GN&C Pointing
- GPS Time Function testing

The testing will evaluate command and data content and timing across the interfaces. The testing will ensure proper functioning of the US GN&C to US C&C interface. The moding tests will consist of all tests necessary to ensure the proper acceptance and execution of all GN&C moding commands and moding scenarios. These tests will ensure that legal and illegal moding commands emanating either automatically or from the crew or ground are properly implemented by the US and RS GN&C and the US and RS C&C computers. The test configuration for these tests will include the US GN&C and C&C MDM/MATE and a math model representation of the RS GN&C and C&C functions. An additional objective of this testing is to conduct early US GN&C to RS GN&C moding testing prior to joint testing in Moscow.

The USOS requirements testing will consist of compliance testing of specific USOS GN&C requirements. The objective of this testing group is to ensure the US GN&C system performs as per the requirement specifications. An additional objective of this testing is to provide signature data and test procedure check-out to the Software Verification Facility (SVF) to enable SVF to optimize formal verification.

The fourth major set is integrated USOS/RS GN&C and C&C system testing and their common interfaces, and consists of:

- USOS/RS GN&C – Stage 5A testing
- USOS/RS GN&C – Stage 8A testing
- USOS/RS GN&C/C&C Nominal Operations testing
- USOS/RS GN&C/C&C Off-Nominal Operations testing

The testing will evaluate command and data content and timing across the interfaces. The testing will ensure proper functioning of the US GN&C to RS GN&C interface. The moding tests will consist of all tests necessary to ensure the proper acceptance and execution of all GN&C moding commands and moding scenarios. These tests will ensure that legal and illegal moding commands emanating either automatically or from the crew or ground are properly implemented by the US and RS GN&C and the US and RS C&C computers. When available, the actual RS GN&C and C&C computers will be used.

The ISS System requirements testing will consist of compliance testing of specific System level requirements. The objective of this testing group is to ensure the GN&C system (USOS, RS, USGS, and

RGS) performs as per the requirement specifications. An additional objective of this testing is to provide signature data and test procedure check-out to the Software Verification Facility (SVF) to enable the SVF to optimize formal verification.

Beyond Risk Mitigation

It is obvious that the investment in the GITF makes it a unique facility in the world and its capabilities to support the ISS Program can extend beyond its initial mission of mitigating technical risk in the integration of the GN&C and C&C systems. There are four high potential contributions that the GITF can provide:

1. Verification support
2. Crew training support
3. Real-time mission operations support
4. Sustaining engineering

Verification Support

Verification activities can be supported in three ways. First, the GITF can provide signature runs to support formal verification the SVF. This is particularly important since GITF signature runs will have the benefit of having real flight software and real or emulated hardware-in-the-loop. Secondly, GITF could be a provider of certified math models to the SVF or provide the certification function for the math models representing the hardware elements. Finally, the GITF can off-load the formal verification facility to allow off-line troubleshooting, thereby reducing the impact on SVF scheduling.

Crew Training Support

The advantages of having real or emulated hardware and real flight software again allows for signature runs and a source of certified models to the Space Station Training Facility. Additionally, since the hardware is an integral part of the facility, the opportunity presents itself for crew members to see and operate the equipment.

Real-Time Mission Operations Support

The high-fidelity nature of the GITF provides the capability to perform mission operations support for system troubleshooting and procedures development. This capability could provide a major safety enhancement to the operation of the ISS.

Sustaining Engineering

Since the ISS will be an operational vehicle for decades, it is a certainty that upgrades will be made to increase performance, reduce operations costs and mitigate obsolescence. The GITF can provide a ground facility to evaluate new hardware elements and software modifications to the GN&C system.

Acknowledgments

There are many contributors to this paper, and the authors would like to recognize their continued commitment to the success of the GITF and the International Space Station program:

NAME	ORG
John Craft	JSC
Susan Gomez	JSC
Debra Hurdlebrink	LMESC
Jerry Kennedy	TRW
Paul Kramer	JSC
Roscoe Lee	TRW
Rick Loffi	JSC
Matt Ondler	JSC
Sonny Parkman	LMESC
Charlie Partin	LMESC
Jerry Pasco	LMESC
Doug Pesek	MDAC
Chad Rowe	JSC
Mark Schmidt	LMESC
Kirk Shireman	JSC
Jim Simpson	JSC
David Stephensen	JSC
Roy Stokes	JSC
Terri Struzinski	Honeywell
Teming Tse	JSC
Jenny Wagenknecht	JSC
Frank Weaver	JSC
Tim Woeste	JSC
Rusty Yates	JSC

References

Models Description Document for the PG-1 Guidance, Navigation & Control System of the International Space Station, McDonnell Douglas Aerospace Corporation, March 15, 1996, MDC 95H0250B.

GN&C Integration Test Integrated Product Team Execution Plan, NASA Johnson Space Center, March 25, 1996, Draft - Revision H.

GN&C Integration Test Facility Master Test Plan (GMTP), NASA Johnson Space Center, March 10, 1996, Draft - Revision C.

Technical Description Document for the PG-1 Guidance, Navigation & Control System of the International Space Station, McDonnell Douglas Aerospace Corporation, September 25, 1995, MDC 95H0223A.

AIAA-96-3532-CP

SPACE STATION TRAINING FACILITY (SSTF) REUSE OF ROBOTICS ENGINEERING MODEL

Maury Minette
Simulator Operations and Technology Division
NASA Johnson Space Center
Houston, TX 77058

Abstract

Frequently, when a large complex system must be simulated, individual pieces of the system already exist from outside sources as working validated models. Under these circumstances it is prudent to investigate the reuse potential of the existing model or models. Generally, this problem reduces, first, to how well the alien model meets operational requirements in the proposed target environment and, secondly, the difficulties posed by accommodating the reuse model architecture with that of its new host. This paper describes the reuse of an existing engineering robotic arm model in a crew training simulator. The paper begins with a summary of the discovery process in which crew training deficiencies are identified for the engineering model. Next, the split of simulation responsibilities is discussed, followed by an overview of the respective architectures. Reuse solutions are then proposed for three areas of concern: maintenance of time synchronicity, moding and control, and bi-directional data exchange. A synopsis of results is then presented in which the computer loading constraint is addressed, and a technique is examined that should minimize divergence between the natural environment models. Finally, transport lag predictions are made for the end-to-end system response.

Introduction

The Space Station Training Facility (SSTF) located at the Johnson Space Center (JSC) in Houston, Texas is responsible for providing crew and ground controller training for all systems that will be present on the International Space Station (ISS). One of these systems is robotics, which will utilize a Canadian developed seven joint arm known as the Space Station Remote Manipulator System (SSRMS).

Initially, the SSTF had planned to build a training simulation for the SSRMS. However, the JSC Engineering Directorate also had plans in the same time frame to develop an engineering simulation for the SSRMS that would be built with their Trick Simulation Environment. Rather than have the station program absorb the cost of parallel development efforts for both an engineering model and a training model, the feasibility of reusing the Trick model in the existing SSTF software architecture was investigated.

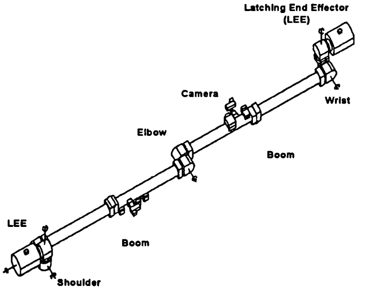
Significant physical differences exist between the two architectures. SSTF models are all in Ada, none of the models run faster than 30 Hz, and all models have to interface and run simultaneously and seamlessly with all the other onboard system models. Moreover, the SSTF uses an executive that employs rate monotonic scheduling and a messaging scheme to exchange data between models. In contrast, the Trick model, which is built in the C language, runs at rates as high as 2000 Hz, uses a global data pool, and is essentially a standalone model. In addition, the Trick model includes its own executive and has its own natural environment model, which includes equations of motion and force and moment calculations.

A SSTF groundrule for the reuse was that no attempt should be made to break the Trick model into pieces and then reuse the individual parts. Previous SSTF experience has indicated that as soon as the "seal" on a model is broken, the engineering effort to retain the integrity of the individual pieces is as great or greater than executing an original design. So, the direction of the effort was to reuse the model *in toto*, or not at all.

Real World SSRMS

The Space Station Remote Manipulator System (SSRMS) is a robotic device that will be used primarily for handling large payload objects on the space station. Payload handling functions include capture, maneuvering, positioning, berthing, deberthing and release. An graphic of the SSRMS appears in Figure 1.

SSRMS is a seven joint manipulator with all joints capable of ± 270 degrees revolution. The arm has two joint clusters (one at each end) and each joint cluster is composed of three mutually perpendicular joints (roll, pitch, yaw). The manipulator is terminated at each end by identical Latching End Effectors (LEE). This configuration allows either end of the SSRMS to be attached to a power data grapple fixture that can act as the arm base. One joint cluster acts as the shoulder while the other acts as the wrist of the manipulator and between the two there is an elbow joint. There are equal length booms between the shoulder and elbow and between the elbow and wrist; each boom is approximately 280 inches long.



All joints shown in zero degree position

Figure 1 SSRMS

The SSRMS has redundant Arm Control Units (ACU) that provide the electronics and software necessary to control the manipulator. The ACU performs arm management along with the kinematic transformations necessary for command purposes. The ACU commands each joint via a dedicated Joint Electronics Unit (JEU), which is a processor with motor control software. Additionally, the ACU commands each LEE via a dedicated LEE Electronics Unit (LEU), which contains a processor with software to control the LEE mechanisms.

Astronaut control of the SSRMS is via the Robotics Workstation (RWS). Data communication between the RWS and the ACU and between the ACU and the JEU and LEU is via 1553 buses. The maximum force in any direction that can be applied at the SSRMS tip is 225 pounds and the maximum moment about any axis is 2285 foot-pounds.

The SSRMS video provisions includes, on either side of the elbow joint, a Video Distribution Unit (VDU), camera, pan tilt unit (PTU), and floodlight. In addition, a VDU, camera, and floodlight are mounted in each LEE. The SSRMS video equipment is compatible with NTSC, EIA-RS-170 and EIA-RS-170A television format standards.

SSTF Requirements

The SSTF will provide the top layer of space station training at the Johnson Space Center. Computer based training on workstations and in part task trainers prepare crew members for the transition to the SSTF, which is a full task trainer. Full task training means that high fidelity simulations of all onboard systems run simultaneously in a completely integrated fashion to provide the full vehicle signature. Every activity that can be performed onboard the space station can be rehearsed in the SSTF.

The reuse of an alien model in the SSTF is more than the relatively simple matter of getting it to run on a different computer platform; the model has to run seamlessly with all the other existing models on this new platform. Consequently, where a model stubbed out its interfaces to other systems such as electrical power, communication and tracking, and command and data handling, these interfaces must now be fully integrated with the existing SSTF models.

The first step in the reuse process involved identifying whether the engineering model could provide satisfactory training in its then current form. While the engineering model was designed to provide very realistic rigid-dynamic arm behavior, a number of limitations were identified in the training arena. Foremost among these were the absence of a dependency on an electrical power model, a limited malfunction repertory, absence of a video system model, and an incomplete telemetry list.

Since the engineering model was already targeted as the basis for several other NASA engineering simulations and the deficiencies noted were inconsequential to these other users, the Trick designers were reluctant to make extensive changes in their design for a single customer. Consequently, a compromise was reached that gave to SSTF the design responsibility for the electrical power model and the malfunction models. Furthermore, it was decided that these SSTF models would be allowed to control when corresponding components in the Trick SSRMS model could run. This would be accomplished by having the SSTF models determine the active/inactive state for each SSRMS component, and then pass a status flag for the component to the Trick model. Trick for its part would insert new code logic to examine the status flags and shutdown components for which the flags were false. When other customers who might be uninterested in this new feature used the Trick model, they would be expected to use an initialization file to set all the component status flags to true.

In addition, SSTF agreed to model the VDU's, cameras, PTU's, and floodlights and design a thermal model to fill the voids in the incomplete telemetry list. SSTF also accepted responsibility to insert the missing data in 1553 bus messages before the messages are handed off to the Robotics Workstation (RWS). And, in the case of RWS commands directed to the video equipment, the SSTF design will intercept the command after it leaves the RWS and process the command within the SSTF video system model. Figure 2 shows an overview of SSRMS command and control functionality plus an indication of those portions that will be modeled by Trick.

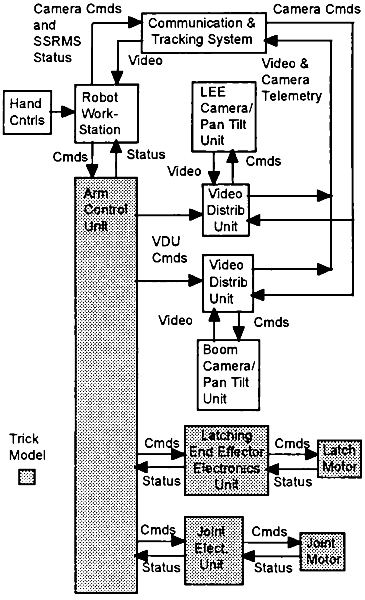


Figure 2 SSRMS Command and Control

Having addressed what enhancements would be needed in the Trick model it was next necessary to decide on an acceptable method for running the model in the SSTF architecture. Before the solutions are discussed it is appropriate to give a brief description of the Trick and SSTF architectures since these were the forcing functions for the approach selected.

Trick Architecture

Trick (not an acronym) is a simulation construction and operation tool that provides code generation services, data and database processors, and user interfaces to help users build and operate specific dynamics and control simulations. References to the Trick SSRMS model are really a shorthand method of referring to the SSRMS model developed using the Trick Simulation Environment. The Trick SSRMS model is a standalone model in that it is first and foremost a robotics model; interfaces and interactions with other space station systems are either non-existent or very rudimentary.

All Trick software is written in the C language and the code, while modular, is not developed in accordance with object oriented design. The Trick executive runs on top of an UNIX operating system and can run the model in either a batch mode or a realtime mode. When the realtime mode is selected the operator must specify a simulation frame size, which is typically 50 msec. Trick relies on an off-line preprocessor to build its job execution file that specifies the jobs and the order of their execution in each 50 msec period (or frame). Job iteration rate and the dependencies of one job upon another are factors used by the preprocessor to build the job execution file. In the realtime mode the executive executes all the jobs that have been scheduled for the current 50 msec frame and then enters a wait state until the next 50 msec, at which time it runs the jobs scheduled for that frame.

The Trick executive uses shared memory for all of its global data requirements. A top level composite data structure that embodies all global data for the simulation is referred to as static memory. Another shared memory segment, arbitrarily sized at one megabyte, is referred to as dynamic memory. The dynamic shared memory segment is used by the Trick dynamic memory management services to provide memory for unconstrained arrays within the static memory segment and to perform dynamic memory allocation at runtime. Both static and dynamic memory are physically allocated together as one block of shared memory.

The Trick model is referred to as a rigid-dynamic model because the flexing of the booms is not modeled (they are considered rigid members) but the dynamics of the joint motors are simulated. This includes such features as motor stiction and hysteresis, which are manifest in control lag and overshoot. Also, the arm exhibits damped oscillations which are a function of how quickly motion of the moving arm is arrested. Trick does not model the Robotics Workstation; instead, analog handcontroller inputs are digitized and an RS232 interface is used to feed the signal directly to the Trick host computer.

Trick provides a functional simulation for the flight software that resides within the Arm Computer Unit (ACU), the Joint Electronics Unit (JEU), and the Latching End Effector (LEE) Electronics Unit (LEU). All the engagement mechanisms (snare, rigidize) of the LEE are modeled. In addition, Trick has a contact detection/constrained motion model for four classes of alignment mechanisms. These classes are the cylinder/rod, v-guide/bar, mating projection/indentation, and concentric discs. The contact detection model depends on the creation of an application specific geometric database involving one of the classes of alignment mechanisms.

SSTF Executive Architecture

The SSTF computer platform consists of two Silicon Graphic Inc. (SGI) Challenge series computers; each computer has been outfitted with eight R4400, 150 MHz central processing units (CPU). The SSTF executive sits on top of the SGI IRIX operating system (OS), which is POSIX compliant. IRIX provides generalized scheduling, basic communication, realtime response to interrupts, and support for other low level OS features. The SSTF executive provides among other services rate monotonic scheduling, message communication between partitions, and moding and control (Run, Freeze, etc.) of the realtime simulation.

Rate monotonic scheduling (RMS) is a scheduling method that uses the rates of periodic tasks as the basis for assigning run priorities to the tasks. The higher the rate at which a task must run, the higher its priority. RMS depends on the pre-load build application of rate monotonic analysis (RMA) tools to the individual tasks. In addition to execution rate, RMA also considers the interdependency of jobs and the maximum execution time for each job. During the pre-load build process, SSTF uses a software tool that automatically sorts models onto CPU's and then performs RMA to determine a workable mix of tasks for each CPU. The RMA mathematically[2] verifies that for the candidate mix of tasks assigned to a CPU there will be sufficient time to run not only the highest priority tasks but enough time left over to run the lower rate tasks. Once the load has been built and the Run mode is active, the tasks are scheduled automatically according to their priority. Highest rate tasks in the CPU are all executed first, then all tasks with the next highest rate, and so forth until all the tasks have executed at their correct frequency. RMS does not specify an exact time within a period when a task will execute but it does guarantee that sometime within the period the task will run. An advantage of RMS is that it allows the execution of models with non-harmonic rates.

Most simulations use global memory to store model state data and the data that must be exchanged between models. For large scale, multi-CPU systems, time consistent and homogeneous data becomes difficult to manage with a datapool structure since models operating in parallel in a distributed architecture will be reading and writing at the same time. The SSTF executive provides message based model-to-model communication. Application models send/receive messages to/from each other instead of relying on a common datapool. Within the same central processing unit (CPU), the tasks can be sequenced such that data produced by one task can be received in the same frame by a task executing later in that same frame. However, because RMS cannot guarantee exactly when within the allocated time period a model/task will execute, the SSTF executive provides buffering for the messages that must cross CPU boundaries. This means that data read by the consumer in one CPU was actually written during the previous period by the producer in a different CPU.

Ada Templates

The SSTF executive architecture depends on the use of Ada code templates to define the structure of a class and the structure of a partition. Object oriented analysis has been applied to the space station systems to reduce each system to a combination of objects that needs to be modeled. In so far as it is feasible the objects being modeled correspond to real world objects; thus a motor might be modeled as a motor object. In addition, more complex objects can be built up from other objects such as combining a motor object with a gear-box object to form a drive_unit object.

The SSTF software architecture defines each object as a class and uses a template to construct the Ada package representing the <object>_class. In the previous example there would be a motor_class package, a gear_box_class package, and a drive_unit_class package, which is formed by use of "with" clauses for the previous two packages. The object_class contains definition for five categories of operations: Create, Initialize, Request_Change, Update, and Selection. The Create procedure is used to elaborate (bring into existence) an object, the Initialize procedure sets the initial state of the object, and the Request_Change procedure provides the capability to insert a malfunction for the object. The Update procedure calculates a new state for the object based on the time step associated with the iteration rate, and Selector functions provide access to data values held within the internal state of the object.

A <model>_partition is an Ada package comprised of all the object classes that make up a space station system or subsystem. The construction of a partition package follows a template format. Every partition defines the following procedures for the model: Set_Up, Create_Data, Self_ Initialize, System_Initialize, Run, Freeze, Hold, and Terminate. These procedures are used to define the state of the model, sequence the model over time, and provide interfaces to the SSTF executive such that the SSTF executive can exchange data with other partitions. (See figure 3.)

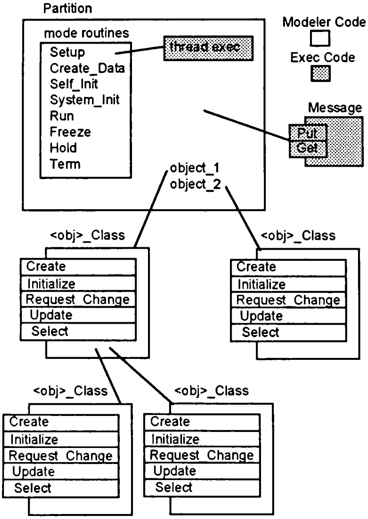


Figure 3 Partition and Class Structure

In order to cycle the model defined by the partition, a thread executive is used. The thread executive is instantiated (brought into existence) from a generic package during the creation of an active training session.

Simulation Control

Overall simulation control is provided by a single Training Session Manager that resides on node 1 but coordinates activities for both nodes through the Platform Manager in each node. Each Platform Manager coordinates the activities in the node by means of a Master Executive that resides in each CPU. The Master Executive for each CPU is responsible for scheduling the Thread Executives, which invoke the desired mode routine such as Run or Freeze. The mode routines call the appropriate model subprograms such as Request_Change and Update. See Figure 4.

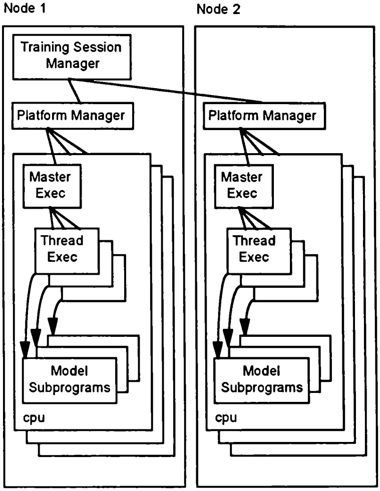


Figure 4 Simulation Control

In a highly simplified example consider the case where the SSTF is in Freeze and the operator inputs a time for going to Run. The operator time input is received by the Training Session Manager which in turn informs the Platform Manager for each node of the time to enter the Run mode. Each Platform Manager then alerts the CPU Master Executives of the run time. When central timing equipment indicates the arrival of the target time, each Master Executive enters the Run mode and begins to schedule the thread executives in the CPU. Each thread executive performs its Run routine, which calls the model Request_Change, Update, and Select subprograms.

Reuse Approach

Because of the unique software architecture in the SSTF the idea of running the Trick model directly under a partition was quickly abandoned. Instead, an approach was decided upon that would restrict one of the SSTF processors to exclusive use by the Trick model. IRIX, SGI's version of UNIX, has command extensions that allow a processor to be dedicated to running a specific process. Three specific steps are involved in the dedication operation: "lock" static and shared memory for process, "mustrun" process on a specified CPU, and "isolate" scheduler for CPU. In addition, the Trick model is given a non-degrading top priority level, otherwise UNIX would decrease the priority of the process the longer it ran. It should be noted that all code executing in the Trick CPU, including interface agents to be described later, will be in the C language.

Next it was necessary to develop a technique that would allow the Trick model to maintain time synchronicity with the rest of the simulation, allow moding and control of the model, and allow data to be exchanged with other models in the SSTF. The foremost requirement is time synchronicity; therefore, it will be addressed first.

Time Synchronicity

Real world time in the SSTF is Greenwich Mean Time (GMT) in days, hours, minutes, seconds, and milliseconds. Simulated Greenwich Mean Time (SGMT) is used also when it is desirable to run the simulation at a time other than the current GMT. In the SSTF, by convention, seconds and milliseconds always agree between GMT and SGMT. SGMT is constructed from GMT by applying the appropriate bias to account for the difference in days, hours and minutes.

The SSTF must run in real time. In a real time simulation the executive must guarantee that the simulated time matches real world time at specified intervals. In other words, the state of a model is calculated for a time interval, then the next calculation of the state does not occur until an amount of real time has passed that corresponds to the simulation time interval. This is different from a "batch" simulation which executes as fast as it can, and there are no constraints to make the state of the simulation progress at the same rate as if object being modeled were operating in the real world.

Central Timing Equipment (CTE) is used to maintain time synchronicity among all SSTF computers. CTE uses the time transmission from the Global Positioning Satellite (GPS) system as its time reference. Each SGI computer has a CTE interface card. Once each second the CTE updates the CTE interface cards with an InterRange Instrumentation Group B (IRIG-B) signal that synchronizes the CTE interface card time to an internal accuracy of 50 nanoseconds. CTE interface card handler software converts this GMT to day of year and number of milliseconds since midnight. This GMT time value is resident in a CTE interface card buffer address and is also stored in SGI shared memory. The CPU Master Executive reads its time from shared memory but the Trick executive gets its time directly from the CTE interface card buffer address. The granularity of the time, which can be read from the CTE interface card, is 1 microsecond.

Rehosting the Trick model from its development platform into the SSTF involved changing the address pointer, used by the Trick run time executive, such that it now reads the GMT from the CTE interface card buffer address. Values for GMT and SGMT are used for moding and control purposes. SSTF goes to run based upon a GMT and enters the freeze mode based on the SGMT time. A discussion of the moding and control should clarify how these times are used.

Moding and Control

The current SSTF architecture dedicates one SGI CPU to running the Trick rigid-dynamic SSRMS model and a different CPU runs the SSTF portion of the SSRMS model. The two halves of the SSRMS model communicate across a shared memory interface. Interface Agents (IFA) controlled by the executive in their respective CPU facilitate the shared memory communication. Each IFA is scheduled to run at 20 Hz. Trick model execution is slaved to the SSTF simulation in that its moding times (when to go to run and when to freeze) are under control of the SSTF portion of the simulation. The overall approach is shown in the Figure 5.

For purposes of this discussion the SSTF has two primary modes of operation: Run and Freeze. In actuality there are other modes such as Initialize and Hold, which will not be considered in detail. In Run the simulation time advances and 50 milliseconds of simulation time is equal to 50 milliseconds of real world time. The Freeze mode stops the advance of simulation time and maintains time at its last value. Initialization is performed in the Hold mode and allows new values to be written into the set of parameters that controls the configuration and behavior of the simulation. Datastore, which also can be active only in the Hold mode, allows a set of parameter values to be captured that defines the current state of the model. These values can be read back in during the initialization process to return the model to its former state. Generally, the same set of parameters is used for Initialization and Datastore.

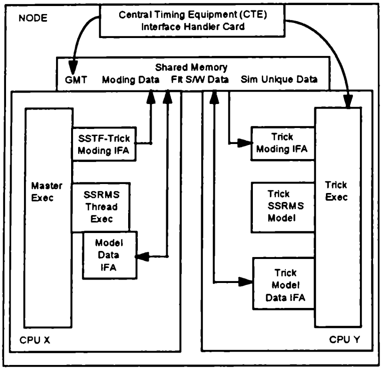


Figure 5 SSTF-Trick Interface Agent (IFA) Overview

The SSTF-Trick Moding IFA is executed at 20 Hz. Each time it runs it checks to see if the CPU Master Executive has requested a mode change. If a mode change has been requested it writes the following information out to shared memory: New_Mode, GMT when the transition is to take place, and SGMT of the future transition. The Trick executive performs mode transitions based on information in its run_input file. The Trick Moding IFA, which is called at 20 Hz, updates the run_input file such that GMT becomes the time for entering the Run mode and SGMT is used as the basis for determining the number of seconds to run before transitioning to the Freeze mode. More specific examples are presented in the following paragraphs.

When the Trick model is in the Freeze mode and the Trick Moding IFA detects a New_Mode equal to Run, the GMT time of transition is passed to the run_input file. The Trick executive continually compares the GMT in the run_input file against the GMT time read from the CTE interface card buffer. When the two times match, a transition is made from the Freeze mode to the Run mode.

The Trick transition from Run to Freeze is similar but is based on SGMT and the number of seconds the Trick Run mode has been active. The Trick Moding IFA kept a record of the SGMT, which accompanied the last New_Mode transition to Run. When the Trick Moding IFA detects a request for a transition to Freeze, it takes the new SGMT, which accompanies the New_Mode request, and subtracts from it the record of the old run SGMT. This delta, in seconds, is passed to the run_input file as the target elapsed run time. The Trick executive continually compares the actual elapsed run time against the target elapsed run time and transitions to Freeze when the two are equal.

Data Exchange

Having established a common time source for the Trick model and a method for moding and control, the remaining step is to establish a mechanism for exchanging data between the SSTF simulation and the Trick simulation. Two categories of data are involved; flight software data produced by the SSRMS computers and simulation unique data. Not all data are transferred at the same rate; flight software data is exchanged at 20 Hz, 5 Hz, and 1 Hz, and all simulation unique data is exchanged at 20 Hz. The data exchange mechanism, however, is the same for both data categories. Controlling the data transfers are a (SSTF) Model Data IFA and a Trick Model IFA, which both run at 20 Hz. Each IFA has a table of export variables and a table of import variables. A double buffer scheme is employed in shared memory to effect the transfer. The buffer size is sufficient to handle the combined set of 20, 5, and 1 Hz data. Each 50 msec is considered a frame and the frames increment from one to twenty. Twenty hertz data appear at the top of the buffer in every frame, followed by 5 Hz data in every fourth frame, and 1 Hz data is at the bottom of the buffer and appears every twentieth frame.

Both SSTF and Trick have agreed to abide by an Interface Control Document (ICD) that baselines the data to be exchanged, the format of the data, the order of data appearance within the shared memory buffer, and the location of data within Trick local memory. SSTF has the responsibility for designing the Trick IFA's and programming them in C. Other than scheduling, the behavior of the IFA's is transparent to Trick because the IFA places the data into Trick local memory addresses defined by the ICD.

One double buffer handles data transferred from the SSTF model to Trick model and a separate double buffer handles data coming from the opposite direction. (See figure 6.) Each 50 milliseconds, an IFA reads data from the half of the buffer that was updated by the other IFA during the previous 50 milliseconds. During this same 50 milliseconds, the first IFA also writes data into the half of the buffer that will be read by its counterpart during the next 50 millisecond period. Also, during this initial 50 millisecond interval the second IFA is doing the mirror image of the read/write activities of the first IFA.

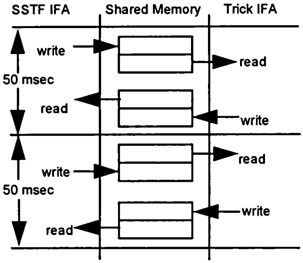


Figure 6 Data Transfer Double Buffer

The buffer address that was written to cannot be read during the same 50 millisecond period because of potential staleness/overwrite problems. A rate monotonic scheduler guarantees that a activity such as write will occur within 50 milliseconds but unlike frame job scheduling it cannot guarantee exactly when within the 50 milliseconds the write will occur. Therefore, if a read were attempted by Trick during the same 50 milliseconds in which the SSTF writes the data there is no assurance that the data has been refreshed.

An interesting feature of the buffer layout arises from the differences between C and Ada. Data structures defined in the C-language always align on the boundary of the largest component. The largest component used by Trick is a double precision variable, which is eight bytes. Therefore, the total size of any data structure passed to Trick must align on a boundary that is a multiple of eight bytes. If C were building the data structure and the components did not add up to this proper multiple, C would pad the structure as necessary to make it fit. However, Ada does not automatically pad. Consequently, when the (SSTF) Model Data Interface Agent is preparing a data structure for transmission to Trick, additional Ada instructions are used to check for a size incompatibility and, if necessary, add the appropriate number of fill bytes to the end of the structure

The Results

Time Synchronicity

For test purposes a software interrupt was created to read GMT time from the CTE interface handler card as soon as the transition from Freeze to Run was made and the results indicate that the typical time difference is about 0.1 millisecond. A similar test was done for the transition from Run to Freeze with an average difference of about 2.5 milliseconds.

CPU Utilization

The initial benchmark when running the Trick Shuttle Remote Manipulator System (SRMS) model in a SGI R4400/150 MHz processor was about 10% CPU utilization. During the development of the Trick SSRMS model the real world presence of a digital servo loop running at 2000 Hz necessitated that certain computationally intensive calculations also be performed at 2000 Hz. Timing studies on the initial release of the SSRMS model indicated that this new model would utilize about 80% of a R4400/150 MHz processor. At this point the anticipated further increase in CPU loading from the addition of the contact detection/constrained motion model forced a reevaluation of the CPU. It was discovered that a SGI node with IRIX 5.3 can run with a mix of R4400/150 MHz and R4400/250 MHz processors. The first benchmark test of the SSRMS model with a single point contact detection model indicated approximately 50% CPU utilization with a 250 MHz processor. Subsequent testing with a four point contact model, which is considered the worst case, indicated a CPU utilization of about 80%. Consequently, SSTF is in the process of upgrading the processor running the Trick model to a R4400/250 MHz card. It should be noted that a node cannot run with a mix of R4400 and R8000 processors.

EOM Divergence

The Trick multibody dynamics model utilizes a vectorized Lagrangian formulation to construct the rigid body equations of motion (mass matrix form) to solve for the states of individual arm segments and the base body. On the other hand, the SSTF environment requires that the motion of all station components be known and then applies conservation of angular momentum to compute the motion of the station as an aggregate. During the early analysis there had been a concern that the natural environment model for Trick might drift from the SSTF natural environment model.

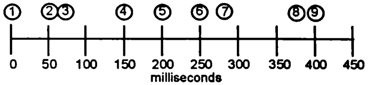
This would be most noticeable during the final stage of grappling a free flying (drifting/unattached) payload. In this situation the visual for the free flyer is being driven by the SSTF environment, but the Trick environment is in charge of the actual capture. Thus there could be a discrepancy between the visual scene and the arm operation. Consideration was given to whether the SSTF environment model should attempt to overwrite the Trick state vector (position and velocity) for the object every 20 Hz. This approach was abandoned for two reasons: the Trick environment model ran at 2000 Hz not 20 Hz and, secondly, test cases run for the two environments indicated only minor drift was present. Consequently, it was decided that when a free flyer entered the capture envelop of the Latching End Effector, a one time update of the Trick environment would be done using values from the SSTF model. During the remaining time to complete capture, the amount of new drift should be negligible.

Transport Lag

SSTF end-to-end transport lag is driven by two factors. For situations when data must be exchanged between partitions in different CPU's, the SSTF executive messaging scheme precludes data computed during one period from use in the other partition until the next period. With 20 Hz partitions this delay is 50 msec. This factor shows up most prominently as a 100 msec delay introduced by the loop-back through the Trick engineering model. This loop-back is mandatory if the engineering model is to run in the SSTF architecture. The second factor is the time delay inherent in creating a visual scene and displaying it. Image generators require a minimum of three frames to create an image; therefore when the SSTF image generator runs at 30 Hz it requires 100 msec to create the scene.

Figure 7 shows the predicted SSTF end-to-end system transport lag for two cases. Step 8 shows that the time between a hand controller input and a change in the out-the-window Cupola view will be approximately 383 msec. Step 9 indicates that the time between a hand controller input and a change in the video scene at the Robotics Workstation will be about 400 msec. Preliminary data from Spar Aerospace Limited[5] indicate the real world transport lag for the same two cases will be about 127 msec and 177 msec respectively. This means the induced transport lag created by the simulation for the two cases is 256 msec and 223 msec respectively. At the present time the SSTF is not mature enough to allow man-in-the-loop evaluation; consequently, it is uncertain whether this amount of simulation lag will lead to negative training.

One possible option to reduce the induced transport lag is to run the image generator at 60 Hz instead of 30 Hz. This will decrease lag by 50 msec but will also result in reduced visual scene content.



(1) 0 msec, Handcontroller (HC) pulse input
(2) 0 - 50 msec, HC input goes to Robotics Workstation (RWS)
(3) 50 - 60 msec, RWS outputs HC data to 1553 emulator card
(4) 100 - 150 msec, SSTF ROB Partition software reads 1553 emulator card and writes HC value to shared memory (SM)
(5) 150 - 200 msec, Trick reads SM and calculates new arm position and writes position back to SM
(6) 200 - 250 msec, SSTF ROB Partition reads SM and creates a message for Visual Partition
(7) 250 - 283 msec, Visual Partition reads message from SSTF ROB Partition and outputs data to visual image generator (IG)
(8) 283 - 383 msec, IG creates and displays image. (Scene change is visible out-the-window from Cupola)
(9) 383 - 400 msec, video frame delivered to RWS and image is displayed at RWS

Figure 7 End-to-End SSTF Transport Lag

References

[1] Simulation Virtual Machine, Kenneth Hill and Rob Sturtevant, Nov. 1994.

[2] The Handbook of Real-Time Systems Analysis: Based on the Principles of Rate Monotonic Analysis, Software Engineering Institute, July 1992

[3] Software Architecture Standard, NAS9-18181, Hughes Training Inc., April 1995

[4} The Trick Simulation Environment, JSC-37943, Revision A, May 1995

[5] MSS Timing and Latency Budget, SPAR-SS-R-0804, Issue E, June 1995

Acknowledgments

1. Tony Nguyen/Hughes Training Inc., for performing the original reuse analysis and proposing the interface agent solution

2. William Hobdy/Hughes Training Inc., for implementing code for the interface agents.

AIAA-96-3533-CP

SPACE STATION ELECTRICAL POWER SIMULATION RE-USE

Mike Belansky
Simulation Systems Engineer
Simulator Operations and Technology Division
NASA Johnson Space Center
Houston , Texas 77058

Abstract

Simulation model re-use is not a new subject in today's technological era, however the methods used for identifying potential re-use candidates, understanding them, and evaluating them in today's distributed business environment are unique. In addition, the typical impediments that exist with traditional model re-use can be hidden by the distributed business environment. The objective of this article is to explore methods used to determine the re-use potential of the Rocketdyne's International Space Station Electric Power System Simulation (EPSIM) relative to the needs of the operations training facility know as the Space Station Training Facility (SSTF). This training facility began its definition phase in the late 1980's and is currently in the final stages of development. Model re-use investigation efforts began in the early 1990's with poor results. Efforts included creation of a model description repository as well as site visits for investigation of applicability. With regard to the SSTF, little re-use commitments were made due to several reasons. The main reason for the non-committal approach can be summarized as non-conformance to requirements. Follow up investigation of re-use potentials occurred. Currently the SSTF is re-using one of Rocketdyne's EPSIM models as a result of a cooperative effort between contractors.

Introduction

Simulation model re-use can lead to significant cost reductions or increases depending on the timing of the decision for re-use. Cost reductions are typically maximized if the re-use is planned prior to or during the application's early development phase. In the event that model re-use occurs after the early development phase, the potential savings is decreased. This is graphically shown in figure 1 where the two unique development costs of each simulation application model are represented by the two dashed investment profiles. This assumes that the development effort begins with the initial overhead associated with building the infrastructure. The bold line represents the summation of costs of independent model development and the line represents the potential cost of development if the decision for re-use is made immediately after the infrastructure has been established. Note that the "break even" point for the re-use decision is actually a function of the cost of integration and infrastructure redesign. If the re-use decision occurs late in the development phase the total cost of that decision can actually exceed the cost of building two unique applications.

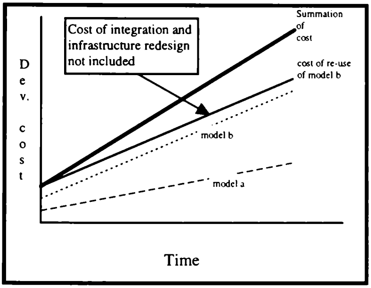


Figure 1: Relative cost of re-use

Re-use benefits

Re-use benefits can exist in the form of savings during development, decreased personnel development overhead, increased product utilization, and decreased out year sustaining engineering expenses. Each of these benefits are tangible and quantifiable on a project specific basis. These benefits will be briefly discussed below.

Development savings: In figure 1 the development cost or re-use is shown to be a function of when the decision is actually made to re-use a model. It also indicates that there is a significant decrease in potential savings if that decision is made after the simulation infrastructure has been established. The greatest cost savings associated with re-use is attainable only at the inception phase of a simulation project.

Personnel development savings: This is an unavoidable expense that is required to educate the model developer in the subject matter of the simulation. This education overhead is a hidden cost that is incurred throughout the simulation model life cycle. In the case of model re-use, it allows for the reduction of number of developers involved in writing the simulation model. The cost savings is the reduction of overhead per developer that is not required to support the project.

Increased product utilization: Product utilization is a means of distributing the development cost over the hours used. As the utilization of a simulation model increases, the development cost for each hour of execution decreases leading to increased product development efficiency. In other words, it is more efficient to use a model of a lesser cost that meets the minimum requirements than one of greater cost and equivalent capabilities. For example, if a lesser expensive model is used to support hardware testing of some arbitrary unit "A" than that of the equivalent unit "B" then the overall cost of the testing unit "A"will have a decreased testing cost due to the lower cost per hour of the test support environment for unit "A" relative to unit "B". This benefit is more evident at the program level rather than the project level. Another benefit resulting from increased utilization is the initial high error discovery rate that is followed by a relatively steep decline in error reporting. This identifies the simulation model problems quickly and therefore achieves facility stability earlier than a lesser utilized facility. Of course this assume that the initial sustaining engineering function is staffed to support the initial increased error discovery rate. This is graphically shown in figure 2.

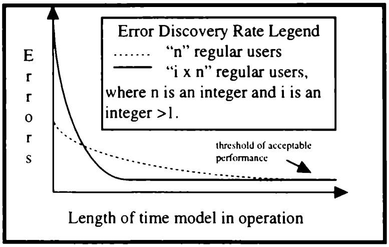


Figure 2: Generalized error discovery rate as a function of number of users.

In figure 2, note that the number of errors discovered by the larger user population decreases sharply relative to that of the smaller user population. The benefit of re-use in this case is the improved user opinion of the model's performance. A benefit as such is quantifiable in dollars through the expense of additional testing and model rework while in the test phase. This would net a user opinion of the model comparable to that resulting from the increased user population.

Decreased out year sustaining engineering costs: As the error discovery rate decreases with the addition of model users and as the back log of reported problems are closed it allows for a reduction of sustaining engineering personnel. This same characteristic is present in the environment of lesser users, except that the point in time where personnel reductions can be achieved occurs significantly further in the future than in the case of the increased model utilization situation. Note in the diagram that the point where sustaining engineering can be reduced occurs when the error detection rate stabilizes. Looking back at the diagram summarizes the significant benefit of model re-use with a greater user population.

Benefits summary: As stated previously each of these benefits is dependent on the specific project being considered for re-use applicability. One of the most significant attributes of a project for which re-use is a consideration is the business environment. The magnitude of the benefits are highly dependent on the business environment. A consolidated business environment has a opportunity for greater pay back than that of a distributed business environment due to elements such as control, communications and alignment of priorities. All these factors must be taken into account when estimating the benefits of re-use. The business environment can also have a significant impact on the associated overhead of model re-use.

Re-use overhead

There can be significant overhead associated with model re-use due to project control, communications, and priorities. Each of these have played a significant role in the planned and current re-use in the SSTF's Electric Power System. Given the estimated overhead, it can still be beneficial to re-use simulation models.

Project control: It became clear that project control is of utmost importance during the investigation phase of suitable applications for re-use. This was one of the underlying factors that kept the Houston, Texas based SSTF project from committing to model re-use from a California based firm during early re-use investigations. In the SSTF's case, management and insight to model development in California was inadequate at the time and too costly to adjust. In addition, the SSTF's application of the model was understood to be secondary to the model developer's primary goal of providing a real-time test environment for the Space Station flight hardware. This overhead was so overwhelming that it prevented the initial re-use commitment from occurring. This draw back may have been overcome by accepting the overhead of placing a liaison in the developer facility to strengthen development oversight and influence.

Communications: Communications is always a consideration when evaluating the re-use potential of a simulation model. Communications is typically relied on to explain planned or actual model capabilities between the developers and the users. Communications are also relied upon to convey the status of the project. A distributed business environment such as that of International Space Station requires extra effort to keep track of upcoming issues and design status. A distributed business environment requires continuous development team communications to ensure project completion and customer satisfaction. This is an additional overhead to all the participating organizations. It requires budget and time to overcome this development hurdle which adds to the cost of development.

Priorities: Task ranking as well politics fall into this category. Local politics are often not understood by program participants of remote locations. In our case, local politics are suspected to have interfered with our initial identification of a re-use candidate. While "communicating" about the California based firm's various plans for model development, we eliminated the candidate that we are currently re-using. This error of eliminating a potential re-use model was caused by poor communications and misalignment of priorities. This error was not caught until three years later in the design of the SSTF. The model that being investigated for re-use was eventually scrapped. The decisions made were based on the SSTF developers' understanding of model capabilities and organizational priorities not the actual priorities of the California based firm. The decision to continue internal SSTF development of the Electrical Power System (EPS) instead of redirecting the development funds and depending on Real time Electrical Power Simulation (REPSIM) model re-use was a safe posture, because the priorities of the participating organizations were not in agreement. Presently in the SSTF re-use efforts of EPSIM have been successful. This success is believed to be the result of aligned priorities that encourage exhaustive efforts by all the involved parties to make the re-use project a success.

Re-use overhead summary: Control, communications and priorities are elements contributing to overhead. These forms of overhead often lead to dependencies that are costly to eliminate at project completion. While integrating the EPSIM model into the SSTF we established a high level of communication with the model developer and the SSTF prime contractor. Upon project completion, the SSTF will transition into the sustaining engineering phase of operations. Moving from a development environment to the operational environment will require the transition of sustaining engineering capability. The cost of such a transition is of course highly dependent on the solution chosen. The SSTF developers are currently investigating the options. A relatively high cost for sustaining engineering was assumed during the trade study phase of the re-use evaluation.

Re-use benefits and overhead summary

This introduction has provided a basis for the remainder of this article by developing the primary concept of re-use benefits and overhead. In the case of the SSTF, three distinct levels of evaluation have occurred. In each of these levels the appropriate attributes will be categorized as either a benefit or overhead. The magnitude will also be estimated. The three distinct levels of evaluation are:

1. candidate identification;
2. candidate investigation;
3. detailed trade study.

These evaluations become progressively more comprehensive. If the initial evaluation known as candidate identification proves to be incompatible, then the remaining evaluations are not performed on that candidate. Each of these levels of evaluation are developed in the following sections.

Candidate identification

Within the Mission Operations Directorate (MOD) of the National Aeronautics and Space Administration (NASA) a team was assemble to perform a NASA wide search of Space Station models that were defined by either NASA requirements or internal support contractor requirements. This team was known as the Model Assessment Team (MAT). This effort was started in early 1990 in an effort to define sources for math models, model performance verification data, and software re-use candidates for MOD facilities. For the purposes of this article, only the software re-use candidates will be discussed.

Evaluation process

The evaluation process required data collection from the model developers and the model users. The data collected from both the users and the developers was intentionally requested as the model requirements to avoid any complications resulting from dealing with the specific implementation of the requirements. Due to this approach, simulation model requirements were reviewed at the component level. In addition, the method of model validation and operational need dates were requested for comparison by the MAT. Additional data was requested from the model developers as to relevant implementation attributes such as the programming language, Operating System, platform requirements, Source Lines of Code (SLOC) counts or estimates, development schedules, execution data and program name. The MAT would identify all the quasi-equivalent requirements for a specific discipline and then call a meeting between the subject matter experts from the user and developer communities to evaluate the difference between each specific requirement. It should be noted that these meetings were telecons due to the distributed business environment of the Space Station Freedom Program.

Evaluation results

This phase of the evaluation process proved to expedite the tedious task of partial model requirement evaluation, but tended to miss the point of total model re-use. This method was successful in the sense that it identified sources to reference for similar model designs, but it failed to identify wholesale models for re-use. It often highlighted the deficiencies and downplayed the areas where the MOD user's requirements were surpassed. This tendency led to the opinion that all the software would require modification to meet the requirements. The expense associated with going into someone else's code and modifying it to meet the SSTF's needs is almost always priced astronomically due to the inherent risks. This is emphasized further by the remote locations of the developers which increased the associated re-use overhead of communication, control, and priorities. The results of this re-use overhead manifested itself in the form of unacceptable risk

Lessons learned

The work performed and products produced by the MAT was excellent in quality, but sanctioned at too low of a level with respect to the NASA organizational structure. The results of this team should have been used to modify existing Space Station Freedom Program level model requirements such that the models fulfilled the needs of the various lower level users. This type of action would have served to align priorities and increase control for the MOD organization. In hind sight, this is quite clear and is actually being demonstrated through a cooperative effort between Hughes Training Incorporated (HTI) and Rocketdyne (RD). The difference in approach between the initial attempt and the current cooperative effort is that the initial approach identified software modifications that the MOD user's developer would have to make. It is suggested that if the MAT had priced the modifications required to meet the MOD user's requirements and pursued funding for the original model developer to upgrade their model's requirements, this team's effort would have maximized model re-use benefits and decreased the re-use overhead through out the Space Station Freedom program. With this twist to the process of the MAT, a cooperative environment is created through incentive.

Candidate investigation

The MAT exercise led the SSTF EPS developers to this phase to further investigate model designs. This initial investigation proved to be unproductive. Another opportunity was made available through the restructured Space Station Program Office to repeat a similar candidate identification evaluation. This time the discussions were raised a level in detail that allowed the perception of a fairly close match with respect to the MOD user's EPS requirements. The programming language, operating system and platformof that match were not compatible with the existing SSTF infrastructure, but it was decided to continue to the next phase to further the SSTF developers understanding of the integration efforts that would be required if model re-use was required as a safety net and also to build alliances with the Rocketdyne Flight Software developers. During the candidate investigation evaluation, there was two way communications as to HTI's SSTF design and development issues as well as RD's EPSIM design, purpose, and schedule.

Evaluation process

The evaluation process began with a visit to the RD facility in Canoga Park, California. As stated previously, the evaluation meeting began with a presentation of HTI's SSTF design and development issues. It was immediately followed by a series of summary presentation of RD's modeling efforts as well as their intended uses. Included in one of those presentations was a detailed presentation of RD's EPSIM. Tours of the RD hardware and software testing facilities were also provided. Emphasis was placed on RD's current use of EPSIM in Flight Software Development and Testing. A detailed hands on demonstration of EPSIM was provided. Detailed discussions of RD's EPS modeling techniques were allocated time in the meeting agenda. A follow-up meeting was arranged in the HTI facility for a detailed requirements review and facility tour.

Evaluation results

The RD tour and openness about their model design and current uses of EPSIM demonstrated their mastery of the subject matter they were modeling as well as building the actual system. RD's day to day communications with the hardware, software, and test engineers were noted several times. Their current ability to support flight software development and testing were discussed. There was a sincere understanding of the SSTF's challenges. Ideas were generated in real-time of how RD could integrate into the SSTF. The requirements review resulted in identification of several EPSIM model deficiencies, but methods of corrective action followed directly. There was a sense of cooperation and need of new business. RD offered a solicitation to become a team member with HTI.

The re-use overhead associated with communication, control, and priorities was addressed indirectly by RD through their actions. Their willingness to address EPSIM deficiencies and potential solutions demonstrated their intent of clear communications. Their willingness to come to HTI's facility in Houston with out funding demonstrated their value of the associated work as well as the priority of the SSTF project. Their solicitation to become a team member demonstrated their intent to share development responsibility and accountability. The re-use benefits that were addressed and recognized as significant strengths were "decreased personnel development overhead" and "increased product utilization". Decreased personnel development overhead is recognized as a strength because of RD's collocation with the hardware and software design engineers. This alleviates the need to educate the HTI developer in this area and the technical accuracy is cross checked by RD Functional Qualification Testing. Increased product utilization is achieved through the use of the same model in the testing, development, and training environment. The re-use benefits that were not addressed at this phase were decreased development cost and decreased out year sustaining engineering expenses. These would be addressed in the trade study phase of evaluation.

Lessons Learned

Perhaps this section should be labeled "Lessons Confirmed". For the sake of fairness, a critical look at all the motivating influences that may have been present need to be understood. It is clear that at this time RD recognized the value and versatility of their EPSIM model, and they recognized the low risk of expanding the models role in the Space Station Program. Development work on the model may have been slowing down and perhaps a down sizing of the support staff was in the near future. RD may have recognized the enthusiasm of their EPSIM development team and wanted to harness their energy. RD's development role in the station was coming to an end with an unsure role in sustaining engineering, and they wanted to ensure a share of the sustaining engineering of the SSTF. RD may have needed additional funding for EPSIM development to meet the final needs of the development and testing role it is tasked with at the RD facility. Given any combination of these reasons, they all lead to cooperation through incentive.

Detailed trade study

This is the final stage of re-use evaluation. This is the phase where the associated model integration options are identified and priced. This is also the phase where all the model deficiencies are identified and priced. The opportunity to identify development, sustaining and operations impacts and associated costs are also accounted for in this evaluation.

Evaluation process

This process began by defining detailed integration options that both the HTI and RD model developers felt they could support. In the case of EPSIM there were four options identified. The next step was to define the criteria that would discriminate the various options with respect to the SSTF's mission effectiveness. The third step was to define weighting factors for each criteria defined according to it importance. The fourth step was to rank each of the options against each other for each of the Mission Effectiveness criteria. The fifth step was to factor in the weighting factors and total the scores. The remaining two steps included costing each option and performing a sensitivity analysis on the results.
In the remainder of this article the various integration options are briefly discussed as well as the mission effectiveness criteria.

Baseline: The first option was the baselined SSTF development approach that did not include the use of EPSIM. The programming language is ADA with a Simulation Virtual Machine executive.

Option A: The second option was to keep EPSIM running in its natural environment which is a DEC Alpha under the VMS operating system along with the Habitat application. The model would still be written in FORTRAN even though the SSTF has been declared an ADA facility. An interface agent would be developed in both the EPSIM's DEC Alpha and the SSTF's SGI host computers to allow data transfer between the two simulation processors. In addition, it was agreed that only a portion of the EPSIM model would be used to reduce impacts to the already established EPS interface definitions with the other simulated Space Station power consumers. This division in the simulation would be placed at the input to the DC to DC Converter Unit which is used to isolated the primary power system from the secondary power system as well as conditioning of the secondary power. This point of division is an arbitrary for the sake of this discussion.

Option B: The third option re-hosted EPSIM in its native software environment into one of the SGI processors. Once again the interface agent would have to be built and the EPS model would also be split to minimize the impact to other simulated Space Station power consumers.

Option C: The fourth option was similar to the hardware configuration of option B with EPSIM running in OSF with an ADA wrapper and transparent Habitat dependencies.

Mission Effectiveness ranking of each option: Criteria used to discriminate the four options relative to the SSTF's mission effectiveness were divided into 4 separate categories. These categories were:
1. technical performance;
2. operational performance;
3. risk mitigation;
4. other life cycle factors.

Technical performance was given 50 as a collective weighting and was based on:
1. meeting the requirements;
2. truth;
3. malfunction capability;
4. completeness;
5. timing;
6. sensor noise;
7. concurrency with flight software;
8. ability to simulate Space Station assembly;
9. conformance to simulation moding;
10. portability to the Space Station Part Task Trainer;
11. the ability to handle P/L changes.

Operational performance was given 30 as a collective weighting and was based on:
1. consistent operator look and feel;
2. ability to accommodate design changes to the vehicle or the simulator;
3. hardware and software maintenance;
4. licensing complexity;
5. complexity of operations skill base;
6. software configuration management;
7. compatibility with the established reconfiguration system;
8. time required to load and execute a training load;
9. time to configure the simulator for operations;
10. extra work and complexity of operations;
11. simulation session re-initialization;
12. Automated Information System (AIS) security compliance due to the SSTF being classified as a national resource.

The risk mitigation category was given 15 as a collective weighting and was based on:
1. schedule risk with respect to completing the EPS model;
2. schedule risk with respect to flight software integration;
3. late approval for option implementation;
4. impact to other simulated power consumers;
5. development cost uncertainty;
6. sustaining cost uncertainty;
7. development risk to the Space Station Part Task Trainer (PTT);
8. development risk to the Russian vehicle simulation;
9. data interface compatibility;
10. design data;
11. computer resources;
12. EPS flight software integration;
13. concurrency with the flight vehicle.

The other life cycle was given 5 as a collective weighting and was based on:
1) RMA;
2) upgrade and replacement costs for both the hardware and software;
3) portability to other platforms.
This category was given a summation of weights of 5.

Evaluation result
The evaluation results that were most desired were the option with the highest mission effectiveness and the lowest development cost. The results are summarized in table 1.

Option	Mission Effectiveness	Remaining Cost
B/L	300	$2.89 M
A	285	$2.74 M
B	334	$2.71 M
C	342	$2.78 M

Table 1: Summary of EPSIM re-use trade study

As can be seen in table 1, options B and C are the best choices, but it should be kept in mind that all the options were relatively close. Option C is preferred due the resulting homogeneity though the technical risks are higher.

Lessons learned

This evaluation shows that the SSTF is approaching the break-even point with respect to re-use of EPSIM. At the time this trade study was performed, the SSTF's EPS model was approximately 70% complete and the infrastructure was in place and operational.

Current status of EPSIM re-use project

To date EPSIM integration work has began in a phased approach. The phased approach first performed the proof of concept which basically implemented option A. This effort has been successful. The SSTF's EPS development team has been reduced from a 5 HTI engineers to a 3 HTI engineers with 2 part time RD engineers. Our next challenge is to implement option C and this effort is targeted for completion in November, 1996. The success of this project must be attributed to the outstanding engineers involved with this project from both HTI and RD. Special recognition must also be extended to the HTI engineers for maintaining positive attitudes during this time of change and increased efficiency.

Mike Belansky
Mail Code DK
2101 NASA Road 1
Houston, TX 77058
Phone: (713) 244-7760
e-mail: mbelansk@gp803.jsc.nasa.gov

AIAA-96-3534-CP

FLIGHT HARDWARE EMULATION IN THE SPACE STATION TRAINING FACILITY

John Stumpf
Simulator Operations and Technology Division
NASA Johnson Space Center - Houston, Texas

Abstract

The Space Station Training Facility (SSTF) will be the primary system used to train flight and ground crews in the operation of the International Space Station (ISS). Fidelity requirements of the SSTF's onboard ISS systems (e.g., Electrical, Thermal, Guidance Navigation & Control, etc.,) simulations require the direct incorporation of ISS Functional Flight Hardware/Flight Software (FFHW/FSW) to form a stimulated simulation of the ISS Command & Data Handling (C&DH) system. The C&DH system hosts the FSW. The Multiplexer-DeMultiplexer (MDM) is the key building block of the vehicle's C&DH system and includes a general purpose processor and ISS unique devices. The ISS Program supports the ground facilities with FFHW MDMs, however, the MDM Emulator is being developed as an affordable alternate to the ISS program unit. This presentation provides disclosure of the SSTF approach for the design and implementation of the MDM Emulator based upon the use of Commercial-off-the-Shelf (COTS) components. At this time, the Emulator has passed proof-concept criteria as an economical replacement for FFHW. Performance requirements for Emulator components do not exceed that of available COTS components.

Introduction

The SSTF has a requirement to run actual ISS FSW in a stimulated-simulation configuration for the purpose of training flight crews, ground controllers, and payload users on the actual flight product. However, there are two main drivers for not implementing a SSTF FFHW subsystem based upon the MDM Functional Equivalent Unit (FEU). These fall into budget and technical categories. For budget, a simulation subsystem structure for the C&DH system based upon the ISS program FEU MDM could not be contained within the SSTF development budget. Secondly, ISS FEU MDM equipment does not support training simulator requirements for mode and control, malfunction insertion, data insight, efficient simulator facility operations, and for integrated multi-facility operations. If these technical problems are solvable for FEUs, they would cause additional budget needs.

This paper provides a brief description of the SSTF's technical approach to develop the MDM Emulator. The Emulator with the FSW models the ISS flight system, however, there are major differences in design and implementation with respect to the flight system. The following several sections of this paper provide the necessary background information for the flight system including a description of a portion of the top level C&DH architecture pertinent to the Emulator design, a lower level description of the MDM component, and a description of the bus traffic messaging.

C&DH System

The C&DH system is a federated data processing system in which the principal computational platform is the MDM. The hardware system provides the platform elements for hosting FSW including the ISS Core and Payload FSW elements. Figure #1 provides a simplified view of the hardware architecture and illustrates a collection of processing-nodes, mostly MDMs, in a tiered bus coupled arrangement. The MDMs are identified by the function each MDM provides. For example, the PMCU MDM provides ISS Power Module Control Unit computational services. (Remaining names are included for reference.) The C&DH architecture reflects physical zone placement, functional organization, and redundancy management design decisions. Three levels of buses service the MDMs and include the Command and Control (C&C) Control buses, the second tier Local buses, and a 3rd lower tier (not shown) bus structure for lowest level data activity. The buses connecting the MDMs are MIL-STD-1553B serial buses featuring a bit time of 1 bit per microsecond. The C&DH system will contain approximately 50 MDMs, each MDM containing a 30386 level processor and other unique devices some of which include processors. Also, Figure #1 identifies other non-MDM hardware. The Portable Computer System (i.e., the Crew Laptop) is the example.

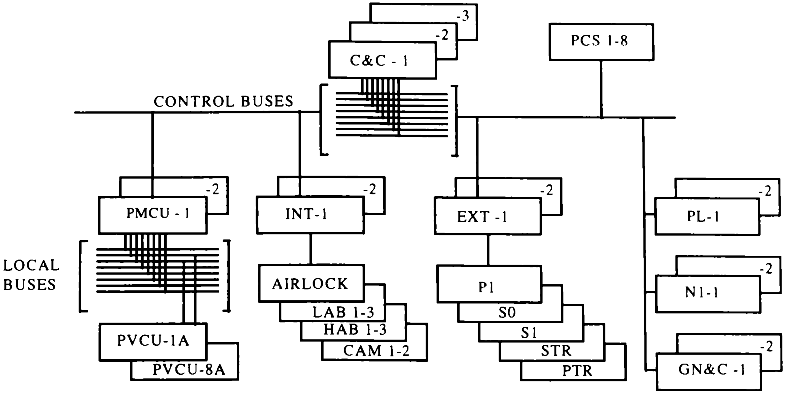


FIGURE 1 - SIMPLIFIED C&DH ARCHITECTURE/1553 BUS STRUCTURE

MDM

The MDM (reference Figure 2) is a C&DH interface to sensors, effectors, and unique devices on-board the ISS. There are two basic types of MDMs which include the Standard type and the Enhanced type. Both MDM types are based upon commercial processor architectures, however, the ISS implementation includes unique components for implementing various data channel types, and capabilities for implementing various Input/Output (I/O) cards. For example, MDMs can communicate over 1553 channels, parallel bit channels and serial bit channels. I/O cards are configured for sensing and for control functions. The number and type of cards are unique to a given MDM. The standard MDM is based on the 12 MHz 80386 processor @ 1.2 Million Instructions per Second (MIPS). The Enhanced MDM is based on the 16 MHz 80386 processor @ 2.2 MIPS. The special outfitting which the MDMs feature for ISS are herein called Devices. The Devices are the core problem for running FSW in Commercial-off-the-Shelf (COTS) equipment and are a major focus in the design of the MDM Emulator.

C&DH Messaging

ISS messaging is a key aspect of the C&DH design for real-time operation. Per the design, messages are scheduled for occurrence within a given ISS subframe (i.e., a 12.5 millisecond time period) and are based upon a contract between a sender and a receiver. There can be a maximum of 15 messages per subframe and MDM processing is scheduled to support the scheduled I/O. Messages may be sent between MDMs at the same level or can originate at a lowest MDM level and be sent up to the top tier C&C MDM. (Also, the reverse is true.) Critical to the message transfer operation is the fact that the MDMs are synchronized and that each message has a time slot on the buses to prevent collisions.

SSTF FHW Emulation Subsystem

In response to the previously stated budget and technical drivers, the SSTF project considered alternate ways of implementing a stimulated FSW simulation. In early 1995, the Virtual Hardware Machine (i.e., the Emulator) approach was selected and a rapid prototyping effort was initiated for the design and implementation. This prototyping effort is being conducted by the Johnson Space Center's Training System Contractor. The remaining sections of this paper provide early design disclosure of the Emulator architecture, a mapping to the ISS MDM architecture and C&DH architecture, and early performance figures.

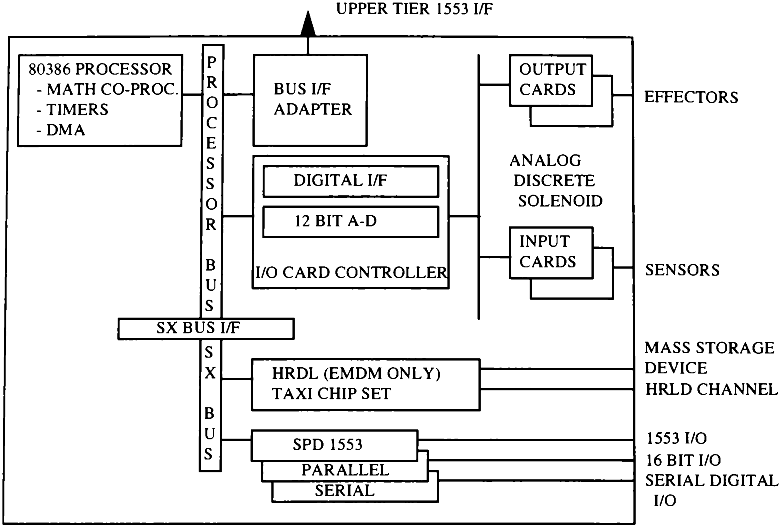


FIGURE 2 - SIMPLIFIED SPACE STATION MDM

The fundamental problem in executing ISS FSW on a COTS Single Board Computer (SBC) arises from the fact that the ISS MDM is configured with special devices not available on the COTS equipment. Not-with-standing binary level compatibility that a COTS X86 may offer for the ISS FSW, any reference to this missing hardware by FSW on a COTS SBC will cause major execution errors. The Emulators role is to enable/force FSW to execute properly on a COTS SBC. This is accomplished by intercepting calls to the missing hardware and substituting device simulations of this hardware to satisfy FSW, thus creating the MDM Emulator (i.e., virtual machine) environment.

Figure #3 provides an overview of the Emulator software structure. Key components include the Kernel, Simulation Executive, and the Device Simulations. These components are supported by the I/O Drivers and the Message System which provides communications to the simulation Host computer and external peripheral devices. Emulator software is hosted on a SBC (i.e., X86) along with the ISS FSW binary image. Note that the Emulator is built on a single fast SBC in order to satisfy all processing requirements which include FSW execution, virtual machine environment execution, and device simulations execution requirements. Each SBC with the aforementioned software comprises a single ISS MDM which can be duplicated and connected to form a larger simulation unique federated COTS structure for the ISS C&DH system. SBC I/O to the simulation Host computer, I/O between MDMs, etc., is implemented through the simulator Host computer VME bus architecture. Unique SBS (i.e., MDM) assignments, as depicted in Figure #1, for the SBCs flow from the FSW binary image that the SBC is hosting. There is one Emulator design which covers both of the ISS MDM types. The following is a more detailed description of Figure #3 components and related functions, and provides necessary background information for a following discussion of Emulator real time subframe execution.

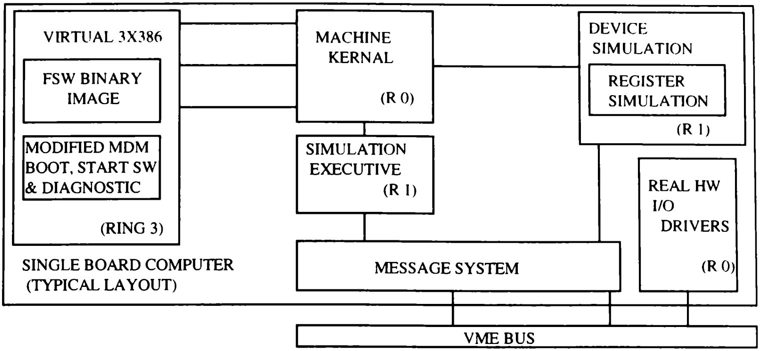


FIGURE 3 - EMULATOR SOFTWARE STRUCTURE

Virtual 3X386: The Virtual 3X386 code section contains two programs which are the FSW binaries and the Start-up code. At the current stage of Emulator development, the FSW consists of a series of modular test FSW (i.e., local code) routines designed to exercise the MDM device simulations in a real-world-way. Each routine works standalone. Start-up code contains the MDM ISS Boot, Start software, and the Diagnostics. In the Emulator, all of this code is considered user code, constitutes a first Ada main, and is placed in processor Ring 3 for processor state protection. Paging for user code is set up to cause faults for accesses to missing device hardware.

Kernel: The Kernel creates and controls the MDM virtual environment for the SBC with the support of other Emulator modules. It is event driven and executes interrupts from other code elements such as page faults from Ring #3. By reason of its location in Ring #0, Kernel can execute privileged instructions necessary for directing the device simulations. For this activity, the Kernel provides processing direction for FSW execution, directs access to a device simulation or performs the simulation itself, and provides memory management services between FSW and the device simulations. Also, it functions to recognize FSW processing errors and reports to the Fault handler. Kernel is in Ring 0 along with the real hardware I/O drivers and the Direct Memory Access (DMA) engine used by the Message System. Kernel is a second Ada main.

Simulation Executive: As the name implies, the Simulation Executive is the key control element of the Emulator. In this capacity, it performs SBC initialization work including determining the SBC/MDM identification and initiating down loads for Emulator configuration information (e.g., FSW) from the host. Other functions include responsibility for Emulator Mode and Control (e.g., load, run, freeze, reset, & data store operations), responsibility for building the Device Simulation Protect Table, Central Timing Equipment (CTE) synchronization, virtual (i.e., the FSW clock) clock control, partition (i.e., program) scheduling, and memory configuration management. One scheduling function is to initiate Kernel activity. Simulation Executive is placed in processor Ring #2 along with the device simulations and the Messaging System. Simulation Executive is a part of a third Ada main.

Device Simulations:

Emulator device simulations simulate the Missing MDM devices of which there are 16. When device simulations are activated during the execution of FSW immediate processing activity is scheduled. Also, Device simulations can be activated by cyclic processing by Simulation Executive. Example (i.e., non-exhaustive example) device simulation scenarios follow:

- Real Time Clock (RTC): FSW reads a time register in the RTC and causes a page fault. Control is transferred to the RTC device simulation. The RTC simulation computes the virtual time for FSW and loads it into a register. Control is returned to the Kernel and to FSW.

- Input Output Card Controller (IOCC): A page fault occurs when FSW tries to read from or write to one of the IOCC registers. The page fault is handled by an interrupt handler in the Kernel and the IOCC simulation is invoked. The IOCC simulation returns data to the Kernel which then returns it to the FSW.

- Serial Parallel Digital (SPD) 1553: FSW creates the data structures to write the 1553 buses and then sets up the 1553 control registers. Because the control registers are on a protected page, this results in a protected page fault and the 1553 simulation is called. The 1553 simulation interprets the instructions and performs the necessary transfers. Upon completion, FSW continues with the next instruction. The current plan is to provide the 1553 I/O and the serial lines. There is no requirement for the parallel buses. The SPD 1553 simulation supports I/O transactions over VME buses and also the physical 1553 data buses such as referenced in Figure #1.

- 82370 Integrated Support Peripheral (ISP): Devices which require the use of an interrupt, timer, or memory access function of the 82370 schedule the interrupt with the 82370 simulation. The schedule time is in virtual microseconds from the start of the current subframe. The Kernel performs FSW virtual to real time conversion.

Other device simulations include the High Rate Data Link, Bus Interface Adapter, Mass Storage Device, Watch Dog Timer, and MDM memory. These devices are planned work for Emulator project completion.

Message System:

The Message system for the Emulator is very different than the ISS 1553 messaging system (e.g., bus structure) partially shown in Figure #1. In the main, the Emulator design uses a VME bus construction for routing both 1553 messages and simulation unique (e.g., control and status) messages. This VME architecture approach is far more cost efficient than implementing numerous serial 1553 data lines. The exception to the VME architecture is for devices such as PCSs and connections to equipment furnished to the SSTF that require physical 1553 data lines. These will be serviced by physical add-on buses. Several aspects of the Message System include Message Memory and Messaging. A discussion of each topic follows.

Message Memory: Message memory for the Message System is a part of the physical simulator Host computer memory. This memory consists of 60 megabytes of space and is allocated on registration basis by the Emulator under control of Simulation Executive. This memory acts as memory mapped I/O through the VME bus and is visible from each SBC in its memory address space. The SBCs can read/write this memory. Similarly, part of SBC memory space is visible to the host as shared memory by reason that each SBC can map its memory out into VME address space. Host memory is allocated during start-up. Additionally, VME space has addressable memory space for physical devices such as timer cards and 1553 cards.

Messaging: Emulator messaging is initially implemented in programmed I/O using the current SSTF Host communication system. During setup, partitions (i.e., programs) register messages with the Message System. The Message System sets up queues and data buffers for I/O noting that messages can be at different rates. In operation, the Emulator has direct participation in messaging (moving) data to/from the Host across the VME bus. One requirement for the Message System is that the messages have to be transferred in the correct subframe. Message latency is determined by the queues. The current Host communication system program I/O approach for messaging is not fast enough for the full Emulator implementation and is an inefficient use of Central Processor Unit resources. The design will be converted to a DMA implementation as part of the final design.

Real HW I/O Drivers:

The Emulator features several Real Hardware I/O drivers required for operation. These include the following:

- SBC Parallel Port I/F to Cable Stub
- VME Bus I/F Driver
- SBC Hardware Clock (8253 Timer I/F) Driver
- Universal Asynchronous Receiver/Transmitter (UART) Driver

In explanation, the Cable Stub provides for hard wired programmed address bits. It is used for SBC address identification. The Cable Stub driver enables the Start software to access these bits and identify a given SBC identity for initial loading and for other activities. The VME driver and SBC Clock driver are code elements required for real-time operation. These code elements support the Emulator architecture features. The UART driver supports a SBC serial I/F used as a debug port.

Real Time Subframe

Figure #4 provides a not-to-scale view of the Emulator's 12.5 millisecond real-time subframe. The subframe layout is not the same as the ISS C&DH subframe, however, the Emulator accomplishes the ISS work. The Emulator's subframe is characterized by the Pre-Processing section, the FSW Execution section, and the Post-Processing section. This contrasts with the ISS subframe where FSW is scheduled to support a stream of 1553 I/O traffic which may occur across the subframe. This is the basis for a design assumption in that the Emulator VME (i.e., 1553 substitute) bus traffic will not be time mapped to the specific occurrences of ISS 1553 bus message traffic events below the 12.5 millisecond subframe level. The assumption is that FSW will be tolerant of any potential time stamp differences between the Emulator transaction time and the FSW/MDM truth time below the subframe level. The Emulator will provide up-to-date and correctly sequenced message transactions at the beginning of each subframe, and will provide up to date processing to support the transactions for the subframe. By exception, the Emulator will provide for exact timing where physical 1553 bus structures are implemented such as in the case of the PCS bus structure. The above is one example of a number of design assumptions. These assumptions are tallied and worked as the project progresses.

Emulator scheduling software in combination with the Central Timing Equipment (CTE) clock (i.e., a facility timing unit) calculates the beginning of the subframe time and kicks off the top of the frame for pre-processing. These activities include some initial processing to save the state of the previous processes, error processing for errors passed from the previous subframe, and execution of the device simulations included any needed I/O. For example, the RTC device simulation may update time registers. Post device simulation processing, the FSW I/F sets up the time slice for FSW. FSW is entered upon receipt of the simulated RTC interrupt and executes in FSW virtual time at a much faster rate then the MDM based FSW. FSW executes until it writes/reads a protected address (i.e., a missing device). When this happens, it executes a page fault to the Ada main containing the device simulation for the missing device. During the fault, FSW virtual time is frozen. After the device simulation executes immediate processing, it returns control back to the Kernel. FSW is allowed to continue executing with FSW virtual time running. The above process repeats for each page fault condition until FSW execution time equals the time allotted at the start of execution. Any time deviation for FSW execution relative to the FSW allotted time is an error condition. If the execution time falls short, then there may be an execution problem in the FSW. If the time is too long, then the Emulator may have an error condition such as a lengthy device simulation. For this condition, a processing error is reported to the Pre-processing section of the next subframe for closure. Post FSW processing, the Emulator provides for 1553 I/O across the VME bus and in any remaining time provides ancillary processing for such things as subframe metrics. For the above, the Host and the BC/MDMs are kept in synchronism through the use of the CTE clock. The clock pulses the Host plus timer cards in each of the VME chassis for time updates. The Emulator synchronizes the SBS timer with the VME timer card.

Hardware Architecture

Figure #5 provides a partial view of the Emulator hardware architecture. The architecture is structured about the VME based SBC and is configured into chassis assemblies featuring a bus structure similar to the flight system (reference Figure #1). In the Figure, each SBC has been labeled with the target ISS FSW function/software load. VME Chassis #1 contains the C&C SBCs. The chassis for tier two and three are organized similar to the C&DH system structure and can be outfitted with (approximately) 15 SBC cards per chassis. This number is in part based upon the maximum bandwidth capabilities of the VME buses. A VME bus structure provides the linkage between the chassis for memory mapped I/O and for servicing other attachments such as the PCS. For the case of the PCS units, these devices are driven from a VME 1553 card. Other cards include a timer card, and optional test cards.

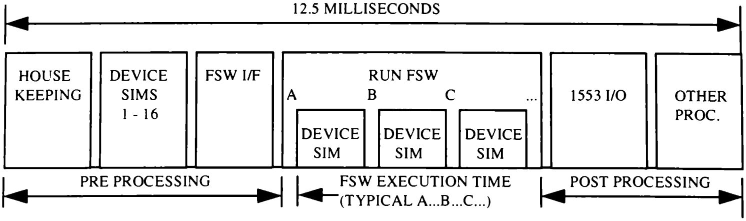


FIGURE 4- EMULATOR EXECUTION SUBFRAME

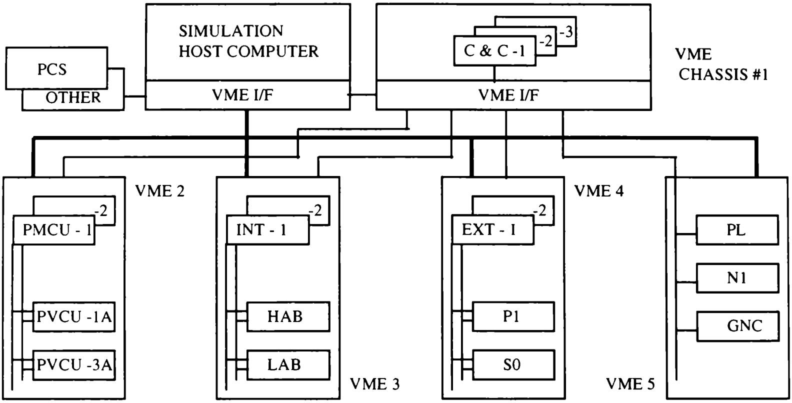


FIGURE 5 - EMULATOR HARDWARE SYSTEM

Development Environment

Figure #6 provides an overview of the Emulator development environment. The environment is structured about two different processor systems and their unique development environments. These are the SSTF simulation Host system and the system that has been selected to support ISS development. Each system features its own compiler/assembler capability and its suite of tools. The simulation Host system affords numerous terminals for code development and a configuration management capability. Emulator loads including the Emulator code and FSW modules are generated and/or modified on the Host system. These code bodies are compiled for initial checks such as syntax error checking and moved to the ISS system. The ISS system provides a cross compile capability for the SBCs and a Probe capability to down load the development code through a SBC serial port. After loading, the Probe capability may be used as a debug tool. The configuration supports simulation Host load development by providing a direct compile capability for the Host test loads. In a mature configuration, the SBS loads will be down loaded from the Host through the VME bus. The development environment also includes a number of test tools including a VME bus analyzer, a 1553 bus analyzer, etc., for debug work.

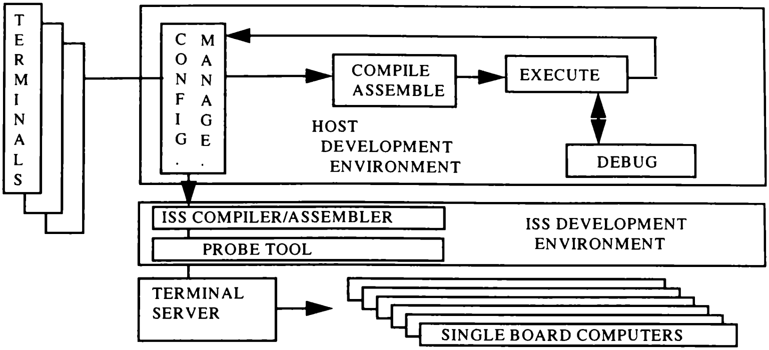


FIGURE 6 - EMULATOR DEVELOPMENT ENVIRONMENT

Status

Project Status

The Emulator is at the Proof-of-Concept project milestone. One product is a cycling software load consisting of the basic Emulator code and a Host test driver. The load performs a message wrap test between the Emulator and the simulation Host. For this configuration (reference Figure #7), the Emulator acts as the bus controller and the Host acts as the remote terminal. The Emulator runs (local) test FSW and exercises key components including Simulation Executive, Kernel, 82370 ISP Simulation, RTC Simulation, and the 1553 Message System. The configuration demonstrates the major design features of the Emulator including page faulting for missing devices, the shared memory design, and other features such as Run-Freeze moding. This is the core capability on which the remaining Emulator system will be built. Also included is a robust set of Technical Performance Measurements (TPM) and analyses which provide insight into system performance. The measurements are of a nature to provide a basis for comparing vender hardware performance and for establishing optimum system performance parameters. For the project, the design work is approximately 80% complete and coding is 50% complete. Key elements of the remaining work include the completion of the device simulations and detailed testing of the Emulator performance against FSW/MDM truth information.

Emulator Performance

Several key results of the Emulator prototype work and TPM/analyses work are that the Emulator Virtual Machine concept for running ISS FSW on a SBC platform is viable and that currently available COTS SBCs and VME bus equipment can be structured to handle the required software functional designs and resulting execution bandwidth. Analysis work indicates that a Pentium level SBC with a clock frequency of approximately 166 MHz will be required for the C&C SBC emulation. SBC loading for the standard MDM emulation is approximately 70% of the C&C SBC and depends upon the I/O configuration. Also, analysis indicates 15 SBCs would load the target VME bus structure to a 50% - 70% level, worse case. Of interest, is the contrast between the ISS MDM processor speed (i.e., 16 MHz clock frequency for C&C) and the 166 MHz clock frequency required by the C&C Emulation. One reason for the magnification lies in the fact that the COTS processor architecture is not fully applicable to the virtual machine task.

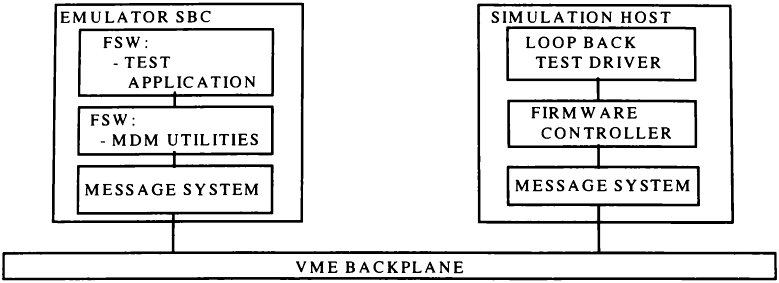


FIGURE 7 - COMMUNICATIONS TEST CONFIGURATION

An overall assessment of Emulator performance is not possible by reason of project status, however, the outlook is positive. For the mature Emulator, performance will be measured against overall required capabilities and includes such items as: does the Emulator provide bit-for-bit or bit-level compatibility with MDM truth information, can the Emulator run unmodified FSW, can a crew member patch the FSW, etc.

Plans

The rapid prototype task for Emulator development is proceeding into the final design phase. Current activity includes the final design of the hardware system, parts procurement, and completion of the Emulator core components including remaining device simulations. Some difficulty is anticipated in the device simulation area by reason of inadequate documentation of the planned use of the MDM by FSW. For this case, available data will be augmented with truth information gathered from laboratory MDMs. One outstanding problem is that of reducing SBC processing bandwidth requirements. This will be accomplished through modifications to the Emulator design and code optimization. Current project schedules call for the Emulator to be ready for Flight Software Integration with the SSTF simulation models on/about 12/96.

Acknowledgment

The Author wishes to acknowledge the efforts by the Johnson Space Center Training System Contractor (TSC) for the design and development of the MDM Emulator. The project team consists of approximately 20 skilled engineers each providing quality project and technical level products.

References

[1] ISSA MDM Emulator Vision Statement
[2] ISSA Emulator Design Review #1
[3] ISSA Emulator Design Review #2
[4] Technical briefing information from the Emulator project staff.

About the Author

John Stumpf is a simulation Systems Engineer in the Simulation Operations and Technology Division at the Johnson Space Center (JSC), Houston, Texas. In this position, he conceptualizes requirements, design, and directs the implementation of training simulators to support ongoing programs such as the current Space Station Program. He has worked for NASA for 33 years and has been involved with simulation systems for most of those years. Current activity involves the implementation of the Space Station Training Facility at JSC. John holds a Bachelor of Science degree in Electrical Engineering from Texas A&M at Kingsville and is a registered engineer in the State of Texas.

AIAA-96-3535-CP

SIMULATION TECHNIQUES FOR AVIONICS SYSTEMS AN INTRODUCTION TO A WORLD CLASS FACILITY

.ed D. Roe, Dan W. Mitchell, Bob M. Linner, and David L. Kelley
National Aeronautics and Space Administration
Marshall Space Flight Center
MS: EB44
MSFC, AL 35812

Abstract

Recent NASA and industry studies have concluded that future spacecraft must substantially reduce annual operating costs while improving system reliability, safety, operability, and capability. These requirements have spurred recent technology developments in avionics architectures, hardware, software, and guidance, navigation, and control (GN&C) algorithms. The ability to test avionics components and systems early in a realistic environment is crucial to the maturation and evaluation of these technologies for use on future and existing spacecraft. The Marshall Space Flight Center (MSFC) has developed a unique avionics simulation capability through the integration of several MSFC avionics testbeds. This integrated testbed consists of three separate facilities interconnected by fiber optic networks for real-time data exchange. This testbed provides the ability to simulate a complete autonomous spacecraft mission that includes pre-launch operations, launch, orbital maneuvers, rendezvous, docking, and landing. The three facilities that support this capability are the Marshall Avionics System Testbed (MAST), Flight Robotics Laboratory (FRL), and Contact Dynamics Simulation Lab (CDSL). Each facility has unique capabilities to simulate specific portions of a spacecraft mission and this paper will present the capabilities of each. These facilities were reviewed by the Vice-Presidential National Facility Review and found to be World Class with unique capabilities. The facilities described are fully operational and are being used to support several advanced NASA initiatives such as the X33 program, Space Station, and Automatic Rendezvous and Capture (AR&C) program.

I. Introduction

The Marshall Space Flight Center has a long history in the successful use of simulation laboratories in the development of avionics systems for numerous launch vehicles. Historically a dedicated simulation lab would be developed beginning in the early phases of a program to support early concept verification and would evolve to become a high fidelity avionics integration facility. In addition to these dedicated program facilities, many labs have also been built for the test and verification of individual components and mechanisms. It has long been recognized that the integration of a number of these labs would provide substantial benefits to both existing and future programs. Potential benefits to these programs include:

- Lowering up-front costs for concept verification through use of existing labs,
- Mitigate risks of applying new technology,
- Able to perform early end-to-end system testing,
- Provide capability typically not available for small low cost programs.

Recent developments in high speed, high bandwidth commercially available fiber-optic networks have allowed MSFC to begin "tying" several avionics laboratories together. The diagram in Figure 1 depicts the current set of labs for which interfaces have been developed. These labs were selected for integration due to their unique abilities to simulate various phases of a spacecraft's mission. Thus, the MSFC can provide a capability to test an avionics system in a mission environment that includes pre-launch operations, launch, orbital maneuvers, rendezvous, docking/berthing, and landing.

The approach taken to integrate the labs has been to use commercial fiber-optic reflective memory products. These reflective memory products have several unique features which facilitate easy movement of simulation data. Some of these features include:

- Low data latency,
- No software overhead to establish communication or send data,
- Ability to generate an interrupt on any or all network nodes,

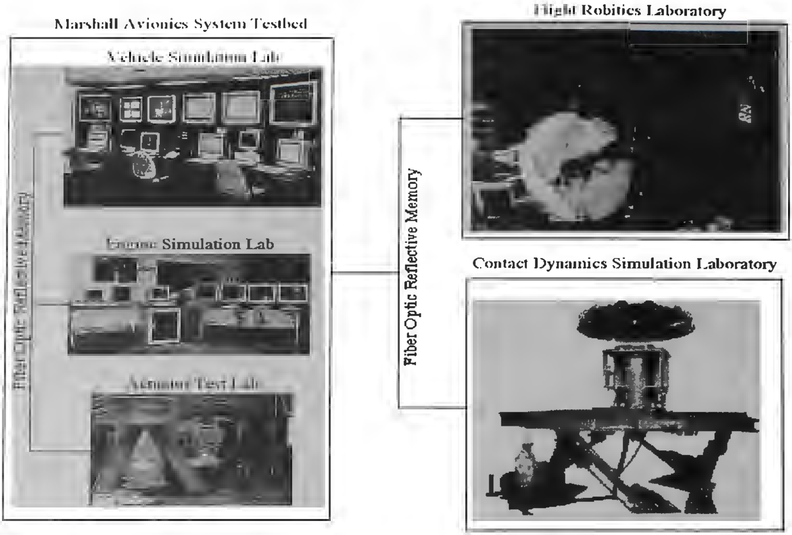

Marshall Avionics System Testbed
Vehicle Simulation Lab
Engine Simulation Lab
Actuator Test Lab
Fiber Optic Reflective Memory
Fiber Optic Reflective Memory
Flight Robitics Laboratory
Contact Dynamics Simulation Laboratory

Figure 1. Integrated MSFC Avionics Simulation Laboratories

- Communication capability over very long lengths, i.e. greater that 2 kilometers,
- Multiple computer platform availability.

In addition to the high speed fiber-optic network, the labs are also linked via Ethernet. This network is used in the transfer of data to real-time displays, in the development of software, and in laboratory command and control. The ability to access the network from the internet allows development and some testing to be performed by off-site users.

The following sections will describe in more detail the labs that have been recently integrated.

II. Marshall Avionics System Testbed

The focus of the MAST is on providing a facility to demonstrate, test, and evaluate advanced earth-to-orbit avionics systems and components such as:

- Avionics architectures and components,
- Guidance, navigation, and control algorithms,
- Health management algorithms,
- Software development methodologies.

The primary objective of the MAST is to provide a facility which programs can use to bridge the gap between technology development and technology implementation. The goal is to make this capability available during early program development and also to allow multiple programs to use the lab simultaneously. Some of the benefits the use of the MAST provides a program include the following:

- Early establishment of realistic design requirements and performance specifications,
- Early resolution of technical risk areas,
- Evaluation of competitive designs through test and demonstration in a neutral setting.

Description

The MAST has recently evolved into an integrated set of simulation laboratories. The labs that makeup the MAST, Figure 2, are the Vehicle Simulation

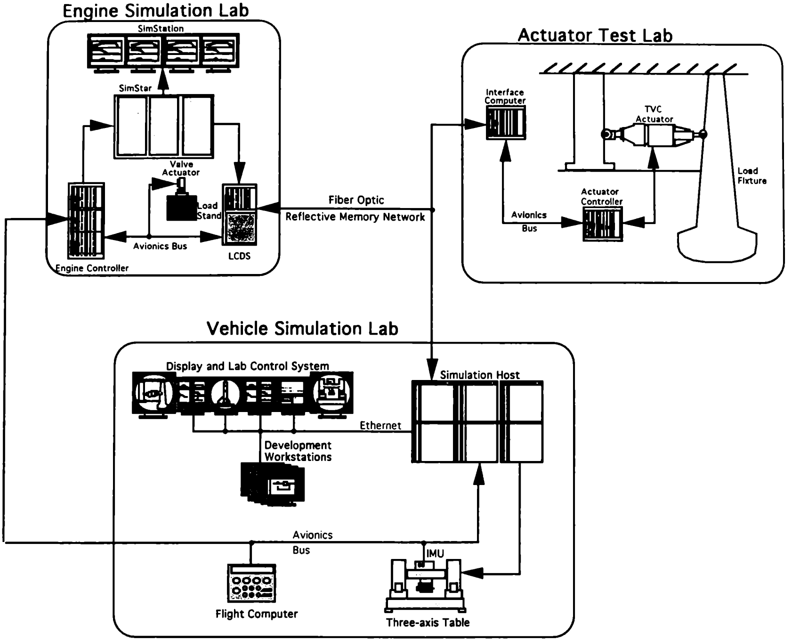


Figure 2. MAST Block Diagram

Laboratory (VSL) , Engine Simulation Laboratory (ESL), and the Actuator Test Lab (ATL). This integrated set of labs provide the capability to integrate, in a closed-loop environment, an end-to-end avionics system required for earth-to-orbit launch vehicles. The capabilities include detailed simulations of launch vehicles, engine systems, environments, and ground systems plus the ability to stimulate various sensors and actuation systems. The labs are designed in modular fashion so that simulations of missing avionics components can be "plugged-in" to the environment. The three labs are interconnected by VMIC fiber-optic VMIVME-5576 Reflective Memory Boards located in each facility. Each board represents a node on the network and is linked by two fiber-optic cables for passing address, data, and interrupt information. Data written to the on-board RAM of any node will appear or be reflected in the RAM of all other nodes. Running at full speed, the link can support a 6.2 Mbyte/second transfer rate. A redundant transmission mode of operation is supported to ensure that critical data is received error free. The amount of time to transmit data from one node to another can be calculated based on the distance between each node and assuming a typical latency of 1.5 microseconds per node.

Vehicle Simulation Laboratory

The VSL is designed to provide a tool for the demonstration of advanced vehicle avionics technologies such as the following:

- Flight computers,
- Navigation systems (e.g. IMUs, GPS receivers, etc.)

- Fault tolerant components and architectures,
- Autonomous GN&C algorithms,
- Automated software generation and verification and validation products.

The major components of the VSL are:

- Three Encore 91 computers hosting real-time execution of detailed vehicle mathematical models,
- Contraves/Geortz three axis rate table, Figure 3, driven by outputs from the vehicle model,
- Display console for real-time display of simulation data and ground station emulation, Figure 4,
- Cabling to support fiber-optic and copper redundant avionics busses,
- Workstations with toolsets (e.g. MATRIXx, Dataviews, G2, TAE+, etc.) to support flight software development, mathematical model development, and data visualization.

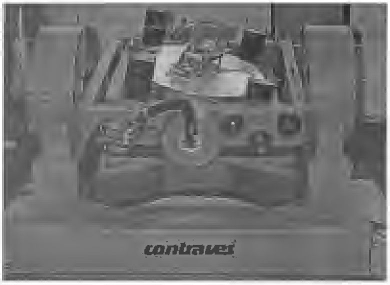


Figure 3. Three Axis Rate Table

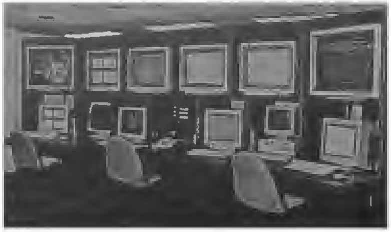


Figure 4. Display and Control Console

Engine Simulation Laboratory

The ESL consists of high fidelity real-time simulations of rocket engine systems with models of high frequency pumps, combustion devices, propellant lines, actuators, valves and sensors. The lab contains both an EAI SimStar and an ADI RTS for the execution of engine models. The SimStar is a hybrid computer that contains both a digital and an analog processor. For execution on the SimStar, a model is partitioned such that the more demanding high frequency portions are executed on the analog processor and the slower portions are executed on the digital processor. The SimStar also includes a real-time display station, Figure 5, for the graphical display of simulation data. The ADI RTS is a PowerPC based platform used in simulation of real-time models. In addition to it's simulation capability, the RTS is equipped with hardware to electrically simulate typical sensor interfaces such as pressure transducers, thermocouples, flow/speed sensors, and RTD sensors.

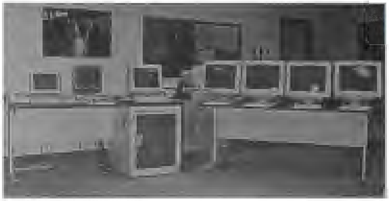


Figure 5. SimStar Display Station

The lab also includes a load fixture to which engine propellant valve actuators can be mounted. The load fixture is used to simulate the loads an actuator would see while operating propellant valves. The load fixture, which is hydraulically actuated, generates loads based on outputs from valve flow models or commanded static loads.

The Laboratory Command and Data Simulator (LCDS) functions as a simulated vehicle interface to an engine controller through an avionics bus. The LCDS also acts as a node on the reflective memory network for the transfer of simulation data between the VSL and ESL.

The capabilities of the ESL allow for the development, test, and demonstration of the following:

- Engine avionics systems (e.g. engine controllers, valve actuators, and sensors),

- Engine control and monitoring algorithms,
- Software verification,
- Hardware and software integration.

Actuator Test Lab

The ATL is operated by the MSFC Propulsion Laboratory and is used in the design, develop, and test of actuation systems. The ATL contains several load stands that are suitable for actuators ranging from small solenoids to large thrust vector control actuators, Figure 6. The lab is supporting active research in the areas of electro-mechanical actuators and power generation and delivery methods. Within the ATL, an interface computer is used to act as the interface between the VSL and ATL. This computer also interfaces with the actuator controller to send actuator position commands and receive position feedback.

Figure 6. TVC Actuator in Load Stand

Operational Scenario

The elements in the MAST communicate with each other at pre-arranged frequencies that are dictated by flight stability and transient response requirements. A test configuration (refer to Figure 2) that exercises the majority of the labs interfaces might include the following components and simulations:

- VSL
  - – Real-time vehicle model on the simulation host,
  - – Flight computer with vehicle GN&C software,
  - – IMU mounted on the three axis table,
  - – Command and control console configuration,
  - –Avionics bus interfacing avionics to the lab,
- ESL
  - – Real-time engine model,
  - – Engine controller,
  - – Avionics hus from VSL to interface with flight computer,
- ATL
  - – TVC actuator mounted in load stand.

Outputs from the vehicle model are used to drive the three axis table. The IMU will sense the motion of the table and supply that data to the flight computer. The GN&C software will compute the current position of the vehicle and the desired position. To achieve the desired vehicle position, the GN&C will output thrust level commands to the engine controller and TVC actuator position commands to the actuator controller. In the ESL, the engine controller commands the modeled engine valve actuators in order to achieve the commanded thrust level. The real-time engine model will vary it's output based on changes in simulated valve positions. In the ATL, the commanded TVC actuator position is relayed to the actuator controller and the actual position is updated. A complete closed-loop vehicle simulation is formed when the outputs from the engine model and the actuator position are fed back into the vehicle model. These inputs to the vehicle model are then used along with models of the environment. gravity, atmosphere, missing avionics components, etc. to calculate the next set of outputs. During the activity described above, outputs from the simulations and telemetry from the avionics are displayed in real-time at the VSL display consoles.

III. Flight Robotics Laboratory

Background

MSFC's Flight Robotics Laboratory (FRL) was developed to provide a single laboratory in which avionics and robotic hardware and software could be tested in a full 6-Degree of Freedom (DOF), closed loop simulation. The FRL objective was to provide a full scale, integrated simulation capability for the support of the design, development, integration, validation, and operation of orbital space vehicle systems.

Description

The FRL is built on developed technologies such as air bearing floors, servo drive overhead robotic simulators, precision targets, gimbals, 6-DOF mobility units, and manipulator and visual system evaluation facilities. A view of the FRL is shown in Figure 7. The facility is

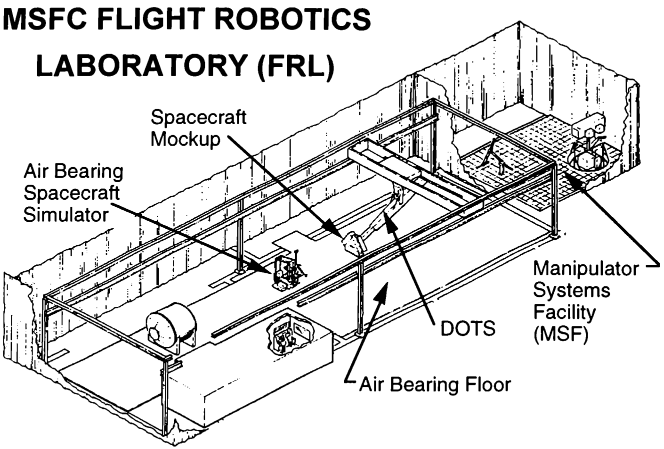

MSFC FLIGHT ROBOTICS LABORATORY (FRL)
Spacecraft Mockup
Air Bearing Spacecraft Simulator
Manipulator Systems Facility (MSF)
DOTS
Air Bearing Floor

Figure 7. Flight Robotics Laboratory Overview

centered around a 44 foot by 86 foot precision air bearing floor, the largest of its kind. A mobility base called the Air Bearing Spacecraft Simulator (ABSS) is used on the air bearing floor and is capable of 6 DOF motion. The ABSS will hold a 400 pound payload. An 8 DOF overhead gantry, called the Dynamic Overhead Target Simulator (DOTS) provides a 1000 pound payload capability for simulating relative motion with respect to a fixed target on the facility floor. A computer system provides inverse kinematics and allows the gantry to act as a target or as the 6 DOF rendezvous vehicle. The target reaction dynamics are simulated through force / torque feedback from sensors mounted at the payload interface. At the east end of the flat floor, a raised floor area houses the Manipulator Systems Facility (MSF). This facility is used for the evaluation and development of robotic manipulators including end effectors.

The FRL also provides a dual RF output, Global Positioning System (GPS) Satellite Radio Frequency (RF) simulator that operates in real time, in closed loop fashion with GPS receiver hardware in the loop. When connected to this simulator, two GPS receivers will provide navigation data as if they were on separate, moving vehicles in Earth orbit, viewing a real GPS constellation.

An observation area is curtained off to allow visitors to view the operation of the facilities during testing. A service area is located adjacent to the flat floor and is used for replenishing of pneumatic and electrical systems on the ABSS. A "jog" panel, located near the computer terminal, is used to switch the DOTS system into either the Manual Mode, Computer Mode, or Idle Mode. A DOTS panic button is located on the jog panel and main cut off switches are located at strategic locations in the floor area.

The FRL is a versatile test facility in that both open and closed loop system testing can be accomplished in either a digital or hardware-in-the-loop mode. It is presently being used to simulate the docking conditions of an automated rendezvous and capture system (AR&C) for the AR&C program. Figure 8 shows the functional relationships of this simulation. The FRL host simulation computer provides interfaces with the articles to be tested. A Night Hawk NH5804 host computer provides the flight vehicle dynamic model that produces dynamic and position data to drive the GPS radio frequency (RF) simulator. In turn, the GPS receiver transmits raw data to the guidance, navigation, and control GPS navigation filter in the vehicle onboard computer (OBC). The OBC interfaces with the host computer to provide the GN&C functions for the simulation. The dynamic model vehicle coordinates are transformed to simulation coordinates that drive the control system of DOTS which, in turn, positions the docking sensor relative to the target pattern. Force and torque sensors transmit contact conditions back to the vehicle dynamics model for closed loop control. Simulation control commands also drive a solar simulation for proper orbital lighting conditions. The dynamic model also drives graphic representation of the simulation and displays dynamic data in real time on Silicon Graphics terminals.

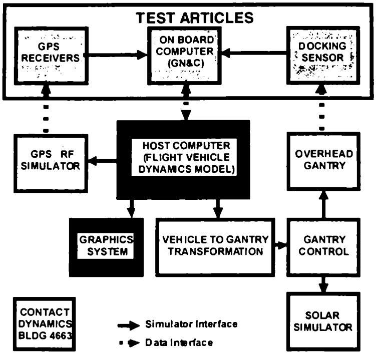


Figure 8. FRL Functional Diagram

IV. Contact Dynamics Simulation Lab

As different space vehicles were developed, interfaces between the vehicles were also developed, creating a need to verify the operation of the interfaces. The Marshall Space Flight Center's Contact-Dynamics Simulation Lab (CDSL) was developed in response to the need to evaluate the Earth-orbit dynamic response of the interfacing mechanisms. The CDSL is a simulation test bed for the study of the contact-dynamics of full-scale docking and berthing mechanisms.

The CDSL allows engineers to simulate how a docking or berthing mechanism would behave in earth orbit under a variety of conditions. This simulation can be used to determine the capture envelope of docking and berthing devices. It can also reveal the stresses a device will experience once in space, through the use of force and torque data recorded during a simulation. Past simulations have resulted in the re-design of some mechanisms, improving their performance, and ensuring that they could perform the task for which they were designed.

Description

A typical berthing or docking mechanism is composed of two mating components, one for each vehicle. A docking mechanism is a kinetic energy assisted mechanism, while a berthing mechanism is a manipulator assisted mechanism. In the CDSL, one component is attached to a motion-base, while the other component is mounted to a force and torque sensor fixed in the support structure above the motion-base (figure 9). The motion-base is a hydraulically driven CAE-LINK Stewart Platform, which is used to simulate the relative motion of the berthing and docking mechanisms in six degrees of freedom.

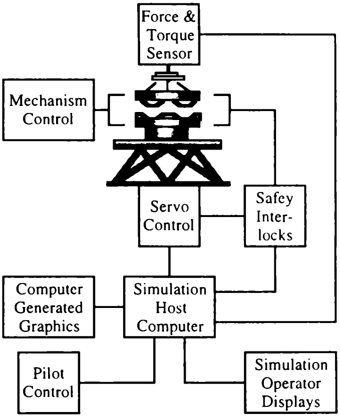


Figure 9. Contact-Dynamics Simulation Lab

The motion-base is commanded by an Alliant FX-80 parallel super computer, which runs a simulation program modeling two vehicles in earth orbit. When the two components of the mechanism touch, the force and torque sensor sends the contact forces and torques to the Alliant computer, which then uses the information to calculate the resulting motion of the simulated vehicles. The computer calculates the relative motion of the simulated vehicles and commands the motion-base to move appropriately. As the motion-base moves, several safety sensors are checked to ensure the test article is not damaged by the motion-base. The computer drives several control panels allowing engineers to monitor the simulation. The computer also sends data to Silicon Graphics workstations to drive graphics displays of the simulated vehicles. Further, the CDSL has the ability to simulate the manual control of a vehicle. A pilot can use control sticks to command a vehicle while observing the berthing or docking mechanism via video cameras or computer generated graphics.

The simulation can be used to dock or berth two distinct vehicles (figure 10), or the simulation can be used to berth two objects together using the Shuttle Remote Manipulator System (SRMS) (figure 11). The two-body simulation has the capability of modeling flexible body vehicles and of modeling different control systems for the vehicles. The SRMS simulation is a complete flexible body simulation capable of simulating the flexible booms and joints of the SRMS, the joint servos, the SRMS control system, as well as flexible payloads and base vehicles.

The CDSL is currently testing the Boeing Common Berthing Mechanism, which will be used to attach International Space Station (ISS) modules together. The modules will be placed in position for berthing using the SRMS or the ISS Mobile Servicing System, which is the ISS robotic arm.

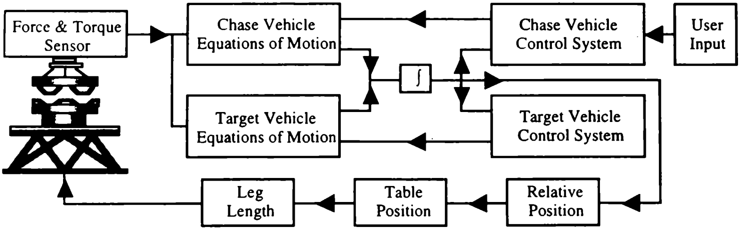

Force & Torque Sensor
Chase Vehicle Equations of Motion
Chase Vehicle Control System
User Input
∫
Target Vehicle Equations of Motion
Target Vehicle Control System
Leg Length
Table Position
Relative Position

Figure 10. Two-Body Contact-Dynamics Simulation

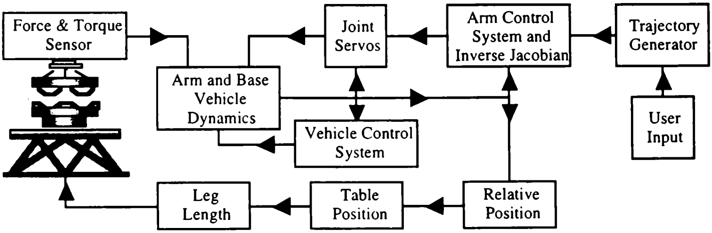


Figure 11. Remote Manipulator System Contact-Dynamics Simulation

V. Conclusions

In 1994, the National Science and Technology Council (NSTC) was directed to conduct an independent review of federal laboratories. The NASA Federal Laboratory Review (NFLR) Task Force was established to review capabilities at each NASA Center. This team determined that the labs described above, the MAST, the FRL, and the CDSL each had capabilities that were world class and unique. This task force defined world class as "a facility that provides a quality, capability, capacity, product, technology, condition, or process recognized by the world aerospace community as among the best in the world." The unique capability to "tie" these world class labs together allows programs to perform extensive end-to-end avionics system testing in an environment not previously available. It is the integration of these individual labs with their unique capabilities that make this facility "world class."

References

1. Roe, Fred D., Kelley, David L., "MSFC Flight Robotics Laboratory (FRL) Description", March, 1995.

2. "NASA Federal Laboratory Review, Marshall Space Flight Center", November 16-18, 1994.

AIAA-96-3536-CP

Rockwell's NAVSTAR/Global Positioning System (GPS) System Test Bed (STB) Aids Block IIF Spacecraft Designs

William G. Burnett*
Rockwell International, Space Systems Division
Downey, California

Dean Wada
Rockwell International, Autonetics & Missle Systems Division
Anaheim, California

Abstract

As the GPS continues to send precise navigational signals to users around the world, the Air Force is considering the development of advanced/follow-on GPS vehicles. Candidate GPS concepts cannot be flight tested but must be validated by simulation/test methods. Issues relating to adaptive guidance, navigation, on-board reprogramability, and control systems, and their related interfaces must be tested in an integrated high-fidelity fashion. The present state-of-the-art in GPS simulation/testbeds is based on the original GPS block II/IIA and has not advanced significantly since the late 1980s. Primary deficiencies are that present methods do not enable real-time GPS simulation/test and cannot accurately describe the space vehicle qualities of the latest integrated spacecraft/payload concepts and related communications/control systems.

A state-of-the-art generic GPS system test bed (Figure 1) is needed to evaluate interoperability (navigational data processing, formatting and generation) with the present GPS system and confirm GPS IIF requirements can be accomplished. Additionally, this system test bed would be used to identify future program requirements and perform preliminary assessments of new GPS technology capabilities.

This paper provides a synopsis of actual system test bed and simulation development operations, and the methodology/operations for a GPS System Test Bed laboratory, including the system architecture of the testbed/simulation itself.

The first GPS's operational flight of the Air Force's new GPS-IIF vehicles is scheduled in 2001, but by that time, engineers will have spent years at its controls and will have flown the new systems through all its mission phases.

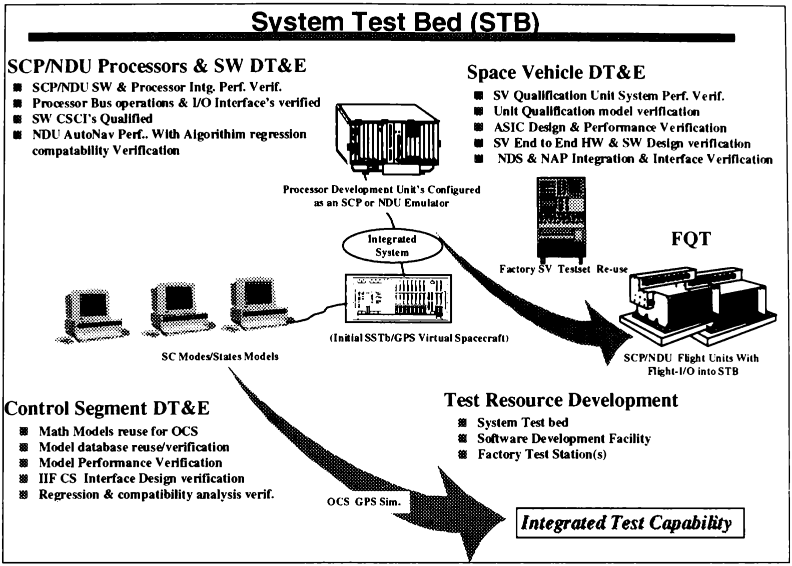
Rockwell has consolidated the major test/simulation activities for its Satellite Operations in its Satellite Systems Test Bed (SSTB) Laboratory located in Downey, California, under the direction of Don Dillihunt. Because the facility is reconfigurable and not built for specific programs, Rockwell is able to modify its configuration to support multiple programs.

1.0 Introduction

Rockwell's laboratory has created test/simulations for such programs as Apollo, Skylab, Apollo/Soyuz, Shuttle, GPS II/IIA, Space Station, B-1B, X31, X33, Brilliant Eyes, EKV, ASAT, etc. and is applying these 30 years of experience to the task of GPS IIF. A secure area within this laboratory has been set aside to support internal research and development. This secure facility is used to test/simulate satellite space vehicles such as the Mars Lander, Mars Global Surveyor, SmallSat, ASAT, etc.. The test/simulation laboratory is designed to support study areas including the following:

- Analysis, operational technologies, demonstrations and direct flight test definitions
- Spacecraft design
- Subsystems integration
- Failure mode analysis
- Instrumentation requirements and definitions
- Flight test techniques and procedure demonstrations
- Test/Simulator sizing and performance requirements definition

*AIAA Member



System Test Bed (STB)

SCP/NDU Processors & SW DT&E

- SCP/NDU SW & Processor Intg. Perf. Verif.
- Processor Bus operations & I/O Interface's verified
- SW CSCI's Qualified
- NDU AutoNav Perf.. With Algorithim regression compatability Verification

Space Vehicle DT&E

- SV Qualification Unit System Perf. Verif.
- Unit Qualification model verification
- ASIC Design & Performance Verification
- SV End to End HW & SW Design verification
- NDS & NAP Integration & Interface Verification

Processor Development Unit's Configured as an SCP or NDU Emulator
Integrated System
FQT
Factory SV Testset Re-use
(Initial SSTb/GPS Virtual Spacecraft)
SC Modes/States Models
SCP/NDU Flight Units With Flight-I/O into STB
OCS GPS Sim.

Control Segment DT&E

- Math Models reuse for OCS
- Model database reuse/verification
- Model Performance Verification
- IIF CS Interface Design verification
- Regression & compatibility analysis verif.

Test Resource Development

- System Test bed
- Software Development Facility
- Factory Test Station(s)

Integrated Test Capability

Figure 1 The System Test Bed Must Provide a Foundation For Early, Incremental System Design Verification & Transition to Subsequent Test Phases

2.0 GPS IIF System Test Bed (STB)

STB's classified laboratory facilities and dedicated assets establishes the focal point for development, integration, test and mission support during the GPS program. The STB provides the flexibility to support an Integrated Test Capability (ITC), while maintaining fidelity and responsiveness to a technically challenging program such as GPS. STB demonstrates early, incremental verification of key performance parameters (Table 1). All space vehicle interfaces and GPS end-to-end system integration testing is supported in this incremental buildup of STB and the related ITC.

The incremental buildup (co-located multiple development and test sites) of the STB was first to support a IIF spacecraft integrated test capability, then with the payloads, establish a space vehicle test capability and finally a GPS IIF system test capability.

Rockwell's test bed infrastructure/heritage has a proven capability, supporting programs such as Ground Based Interceptor, ASAT, Satellite Systems Test Bed (SSTB), and Brilliant Eyes with a common test bed infrastructure. This basic architecture has been applied to new programs such as the X33, the new integrated Propulsion Checkout and Control System (PCCS), and the new Vehicle Automated Checkout (VAC) system for NASA's Space Shuttle.

STB Activities	STB Support Approach
SC ITC Established SCP (Processor and SC Subsystem Integration & Test) Multiple Development & Test Sites	•Dedicated STB assets including SCP VME/R6000 & S/W Programming development Units (PDU's) computers. •Breadboard and engineering model-class spacecraft hardware subsystems configured into STB
SV ITC Established for all EDU's, & FSW FQT Common Processor SCP/Payload Integration & Test NDS (Global Burst Detector -GBD) & Reserve Auxiliary Payload-RAP Intg'd	•Automatic test execution, control, evaluation and reporting •EDU's & QUAL:'s Processors for NAV-NDU/SCP •Responsive data collection and analysis for thorough and detailed testing •EDU/QUAL/Payload GSE Hardware-in-loop testing DOE GBD/RAP to SV Simulated I/F's
GPSIIF Software development, integration (FSW, Sim, OCS, IMOSC, GSE) Navigation payload updates and State-of-Health Monitoring Telecom Simulator Functions & Test	•Multi user Unix Sun, SGI and R6000 workstations software development and simulation environment •Offsite Spacecraft/payload software development and remote STB test monitoring and control including S & L-Band Transmission Cap.. •Provide realistic GPS System dynamic real-time simulations with hardware-in-loop (Includes GA & MS functions).
GPS/Control Systems analysis and design trade study support (Crosslink commanding and user-spcified telemetry)	• Provides GPS Envmts for early autonav algorithm verification and system timing studies • Provide tools to verify requirements, design and interfaces including document development and control
GPS IIF System ITC Established Control System/OCS/IMOSC model/database development Pre and post launch telemetry command validation	• Directly apply STB GPSIIF system & subsystem test/models/databases to Control System/OCS GPSIIF upgrades • STB provides direct interfaces to CS/IMOSC/AFSCN for command testing (Intg. of GFE 'd TS & AUTONAV Emul.)
Ground Support Equipment (GSE)	• Pre-factory checkout and launch software verification/validation
Mission operations analysis support	•Analysis of Control Segment system operability •Command sequencing and timing
Post launch anomaly resolution	•STB configurable to current spacecraft configuration •Data collection and analysis

Table 1. STB Capabilities Grow Incrementally & Progressively From A IIF SpaceCraft to a Full GPS System Integrated Test Capability

2.1 SpaceCraft Integrated Test Capability Established

Networked workstations, data servers, simulation hardware, and System/Simulation Infrastructure (S/SI Version 2.0) software provides multiple users the capability for local or remote test/simulation execution in both real-time or non-real-time modes. Rockwell provides both offsite payload developers and the Air Force Satellite Control Network (AFSCN) this STB software infrastructure. The STB system architecture is comprised of an existing System Support Layer infrastructure, COTS hardware and software components, and simulation models (Figure 2). This architecture provides high fidelity at relatively low cost and risk.

2.1.1 STB System Support Layer (SSL).

This system utilizes a layered configuration of executives, test control, graphical user interfaces, hardware interfaces, fault inject, and data capture and display services. This SSL, proven many times over, is a layer of workstations, COTS operating software, compilers, and debuggers interfacing to standard VME based processors and signal

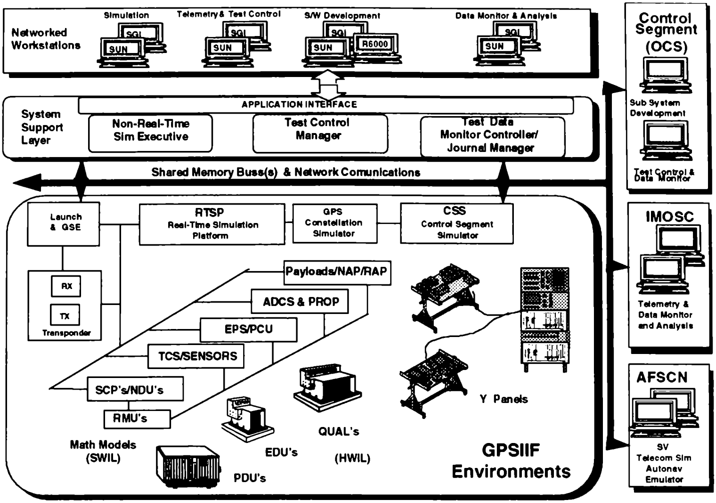


Figure 2 STB System Architecture Supports Initial EDU's Space Vehicle Testing, Migrates to QUAL System Testing and Development/Test of Operational Control System (CS) Software/Models

interface cards. Constant improvements over the last 9 years with active technology enhancements/upgrades have gone into producing this "middleware" product.

The **Simulation Executive** functions provide both Unix based non-real-time and embedded real-time multi-tasking, multi-processor task scheduling, shared memory interprocess communications, and symbol table driven data inject/collect services. Functions including mode control, common time source, save/load conditions, time jump and traps are fully supported.

The **Test Control Manager** provides the single point of test control to all STB lab functions establishing both repeatable reliable quick test execution and control. A common test language, executable in batch mode interfaces with real-time data collection/inject services to insure proper event timing, sequencing and data evaluations.

Test data is sent to the **Test Data Monitor and Journal Manager** functions for monitoring and test report generation. Test Data Monitor Controller and Journal Manager provide systems services and standard simulation user interfaces for data monitoring, inject and test data logging. Multiple data stream collection and correlation for real-time display and post data analysis are provided along with graphical visual scenes of GPS spacecraft, constellation and experiments.

2.1.2 STB Test & SpaceCraft Hardware.

STB utilizes dedicated hardware assets including networked user development workstations, real time simulation and telemetry systems, STB spacecraft hardware, and secure remote user interface communication links. Networked workstations provide a user environment in which all STB components can be easily configured, controlled and monitored (e.g., simulation, Engineering Development Units (EDU's), Flight Software (FSW), Test Manager, telemetry) from a single or multiple workstations in either local or remote operations. Networked workstations provide both native system compilation and target (UNIX, R6000, or R3000) development, debugging, and test.

The Real Time Simulation Platform (RTSP) provides simulation models execution, spacecraft data bus and electrical interfaces, GSE and test hardware interfaces, and simulation test and control network interfaces. The RTSP consists of COTS hardware components including R3000/R4600 simulation processors; BIT3 dual-port RAM, VSB shared memory and VME reflective memory; SBS Mil-Std-1553 interface cards, IRIG common time source, VME discrete, analog cards, and serial telemetry cards.

All telecommunications are designed to fulfill the basic IIF and II/IIA/IIR command and upload protocol interface requirements. These "Telecom Simulator" (TS) requirements provide telemetry, tracking, and commanding capability to send/receive clear or encrypted Space Vehicle commands, process the commands and send/receive back authentication/verification as well as functional status change. The TS must also format and execute these telecommunications via hard-line or RF links and transmit L-band, S-band, and Crosslink messages for reception in the simulated control system.

The **GPS Constellation Simulator** is a multi-channel signal generator (Up to 20 satellites) supporting P & Y Code and having 10 channels of L1 and L2 outputs. The Selective Availability/Anti-Spoof is supported with the required Auxiliary Output Chips (AOC).

The **Control Segment Simulator** (CSS) provides a dedicated telemetry real-time uplink/downlink/crosslink data formatting, acquisition, and monitoring. End-to-End system testing is accomplished using simulated monitoring and ground antenna stations. This unit also provides user interface telemetry command builder functions and is used for CS/IMOSC/OCS AFSCN telemetry/data compatibility testing.

Existing VME Programming Development Units (PDU's) i.e., VME GPSIIF Space Craft Processor (SCP) and NAV Data Unit (NDU) configured with COTS VME boards (shared memory, digital/analog, serial, and Mil-Std-1553) running VxWorks are used in the early FSW phase for embedded processor prototyping and application development.

The SCP/NDU VME brassboards provide functional operations (timing, throughput, interfaces, etc.) as the flight article and are dedicated lab assets. These units are configured with a Peripheral Component Interconnect (PCI) bus interface allowing direct memory access for data collection and data injection capability.

These VME brassboard PDU's are replaced with Engineering Development Units (EDU's) as they become available . These non rad-hardened flight equivalent units support the buildup and complete integration and test of the flight software Final Qualification Tests (FQT's)). The EDU's will have actual physical volumes and wire/cable harnesses.

Once the EDU's are integrated into the STB, acceptance testing with the latest software release commence. This testing starts with unit and proceeds to subsystem and system testing. These risk reduction test sequences precede all "Black Box" integration and testing.

For deployment purposes the STB system itself has been designed to be moved if needed. All test bed systems, power supplies, etc. are rack mounted in VME cages. All related work stations are also portable.

2.1.3 STB Software/Simulation Models.

STB IIF spacecraft/vehicle/system models are developed in Ada or "C" to execute in both real-time (VME/R3000) and non real-time (Unix-SGI/SUN) multi-task simulations. Standard model interface communications layer, execution mode control, and model error and fault insertion libraries schemes are used to enhance model reuse. In addition standard planetary environment models, coordinate transformation and mathematical functions and libraries are available for model application usage.

Maximum use of COTS/GFE'd models and/or software for the IIA/IIR satellites and the GPS IIF subsystems deliverables (Models and/or software/code) is stressed. COTS software is used for windowing, human interface displays, database development, local area network communications, and on-line documentation.

The workstation windowing environments utilize the MOTIF standard for all GUI's. Multiple windows are provided to permit users to manage and view the status of several ongoing activities, thereby improving productivity.

Human interface development utilizes the Data Views from Visual Intelligence, Inc. Data Views has proven to be very robust in its power to develop custom displays and to support user interaction. The product is mature, available on most commercially available UNIX workstations. Dataviews permits the testbed user to customize graphics displays to their specific needs.

Database development utilizes the ORACLE tool and in-house "Flat file" structures for import/export of data and the SQL standard for user data display and report requests.

The testbed Local Area Network (LAN) utilizes the standard IEEE 802.3 Ethernet protocols supporting the Network File System (NSF) and Remote Procedure Call (RPC) open standards promoted by Sun Microsystems, Inc. The use of these standardized network services plays a key role in the access of users to the testbed data. NFS allows transparent remote access to any disk on the network. RPC allows processes to communicate with each other on the same host or from a different host. These commercial network processes allow the local or remote testbed user easy access to all testbed services both in real-time or non-real-time modes.

On-line documentation employing quick-lookup hypertext links of all Interleaf documents in a windowing environment relieves the user of time consuming documentation searches. Hypertext links and hypertext markup language tools are provided by Cyberleaf and window viewing is enhanced by the use of Mosaic by the National Center for Supercomputing at the University of Illinois.

3.0 Space Vehicle Integrated Test Capability Established

GPS activities have already been initiated utilizing Rockwell's SSTB facility. Prototype navigation/ spacecraft flight processors executing in SSTB's RTSP have demonstrated GPS II/IIA backward compatibility. Prototype GPS Navigation algorithms were ported from the Ada software development environment to the embedded VME R6000 processor and executed with èxisting navigation payload/ spacecraft subsystems (BDP, FSDU, clocks, L1, L2, L3, etc.). Figure 3 depicts how the test bed was configured to support an end-to-end space vehicle. The test bed is designed in a modular fashion, allowing software models to be replaced with real hardware (brassboards and/or EDU's).

The block IIF design approach for the navigation data unit was validated through the development of a brassboard. The brassboard consisted of the R6000 processor and a single 6U VME module which supported navigation payload code/data generation, and provided all interface functions for the brassboard. The design was based on the IIA architecture and utilized ASICS and field programmable gate arrays to eliminate parts and increase design flexibility. The block IIF navigation unit brassboard supported the generation of navigation data , performed the anti-spoof (Y code) tasks, supported the formatting and transmission of Nuclear Detonation Detection System data, processed uploads and ground commands, and generated telemetry data. Prototype navigation data unit software was developed. IIA algorithms were coded in Ada with IIR interface compatibility added. The prototype software performed approximately 80% of the functions required for IIF. Interfaces to the BDP, L1, L2, and L3 transmitters, S-Band, and FSDU were supported.

The test bed simulated the virtual space vehicle, GPS ground antenna and master control stations (actual GPS data format/protocol up and down), supplied satellite commanding and state-of-health monitoring functions including the specific navigation uploads. The test bed acting as a "Telecom Simulator", provided overall time synchronization and test control, supplying real-time telemetry monitoring and data collection and displays.

The test bed provides verification of compatibility of the IIF brassboard and later the EDU, Qual and Flight units with the rest of the Block II constellation. By utilizing the Test Bed, rapid integration of the nav data unit brassboard was made possible. Checkout of C/A, P and Y code generation, code slewing, and navigation data is difficult due to the long pseudo-random patterns of the codes. The use of IIA test receivers that are a part of the Test Bed provided quick measurement of code generation, slewing and timing. This feature along with the use of the field progammable gate arrays and modular software allowed a rapid prototype development environment. Producing a working unit in 10 months from start of design including two functional enhancements of capability (Y-code generation, and NDS/BDP) from the original baseline design goal.

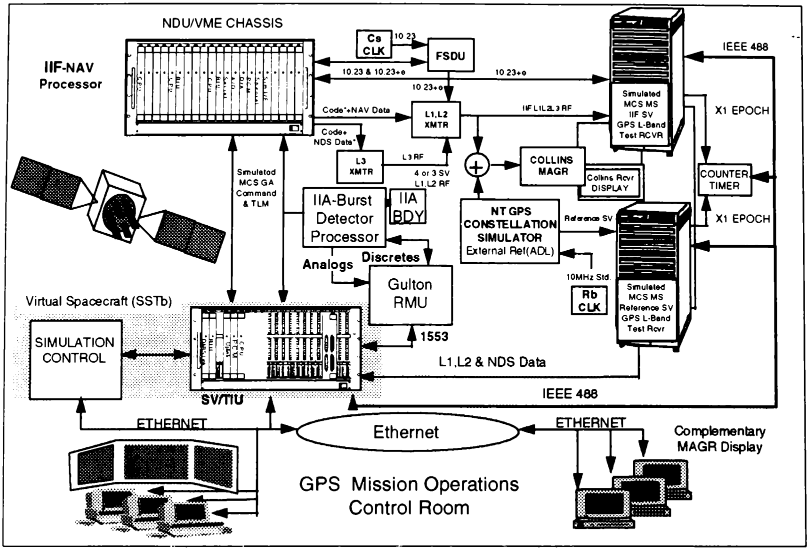


Figure 3 Testbed Configured For End-To-End Integration of The Navigation Payload Specific Interfaces:, The User Segment, Control System, and The IIF Space Vehicle; Demonstrates Interoperability of Payload Functions and Interfaces, and GPS User Receivers

3.1 Typical Test Bed Integration and Test Operations

With this test bed arrangement the Rockwell team demonstrated launch, early orbital operations, navigation payload initialization, and day-to-day orbital operations of a GPS IIF satellite. Rockwell's testbed is based on actual system requirements specifications generated from past IIR and IIA heritage and present IIF derived requirements. All hardware and software have been accomplished using GPS IIF specifications.

A typical test includes a complete 6 Degree-of-Freedom (6-DOF) real-time spacecraft launch. After the last stage of the launch vehicles fires and separates, the GPS satellite is spin-stabilized up to 50 rpm's. The GPS IIF perigee kick motor ignites to circularize the orbit and then shuts down. The GPS satellite despins to zero rpm's. Once you are in the target orbit the satellite's solar arrays are deployed, the Sun and the Earth are acquired. The attitude control then switches from thruster-jet control to reaction wheels. By the end of this phase, the satellite is in its final orbit slot stabilized in three axes and the clocks are turned on. The spacecraft status is then monitored from the simulated ground control segment.

The control system commands a "power up" to the satellite's navigation data unit or NAV processor which starts when the simulated Mission Control Complex hands control over to the simulated Master Control Station. Ground displays now show the nav processor command status and data.

Mission-specific sequences are now commanded to power up the frequency synthesizers and distribution units, the nav data unit, and associated L-Band transmitters. The Northern Telecon GPS constellation simulator is initialized to supply reference GPS RF signals to the GPS user sets. Next, the single-channel receivers that represent the ground monitor stations are configured, and the test bed prepares for simulated S-Band data transmissions of uploads and PCM downlink.

The test bed then commands a processor reset and nav payload initialization, and configures for simulated L1,L2,L3 (includes outputs from the burst detection system) data transmissions to the ground. The nav data unit begins to output navigation messages on L-Band RF signals (L1, L2). Receiver #1 of the monitor station locks onto these signals. Receiver #2 of the monitor station locks on to a similar space vehicle of the reference constellation.

The simulated monitor station picks up the navigation messages and determines the time through the Z_Count and X1 Epoch offsets from the satellite to the reference GPS constellation. Z_Counts and P-chips slew uploads, calculated by the testbed, are issued to the nav data unit to align satellite time to GPS constellation time. The test bed also calculates the fractional part of the P-chip as a clock bias term and clock frequency corrections, which are uploaded as part of the GPS IIF ephemeris. All upload and downloaded data are checked for valid PCM transmissions.

Next the test bed uses a five channel GPS receiver and all the satellites in the simulated constellation (1-GPS IIF, 4-IIA) to solve a navigation problem, in this case the location of our laboratory.

3.2 Test Bed Subsystem Integration and Test Operations

Demonstration of GPS IIF sub functions such as the Burst Detection Processor were easily integrated and tested. Predicted states are determined from reference data from Block II/IIA/IIR systems. This data is then used as truth for the subsequent test. Figure 4 shows a detail integration configuration that was used for testing the interfaces between the spacecraft processor and a related remote multiplexing unit (discretes analog A/D's and D/A's) and the BDP.

To test the GPS BDP system, the test bed commands a power up and uploads/initializes the BDP configuring for a pattern test. The navigation unit performs the data processing for the test, which is downlinked via L-3 and displayed in the simulated master control station. The test bed then issues a mock nuclear burst event by generating a fixed pattern of bits simulating a crosslinked event message. The message is decoded by the BDP in the nuclear detonation detection system and is then downloaded to the test bed via L3 for remote display and evaluation.

Conclusions

The simulation/test bed is designed to explore GPS technology in a way that will expand the vehicle flight envelope from launch to on-orbit operations. The simulation test plan parallels the GPS IIF proposed flight test plan and operational test objectives.

Objectives established in the simulation test plan are expanded as capabilities are added to the simulator. This arises from the role simulation plays as a subsystem test validator and test tool.

The simulation/test bed has sucessfully been used to evaluate GPS concepts well ahead of future contract requirements paving the way for the next phase in the GPS development cycle. Rockwell's GPS IIF Test Bed Team has demonstrated GPS space vehicle and control segment interfaces, related telemetry monitoring and user interfaces to further reduce the design risks associated with the GPS IIF development.

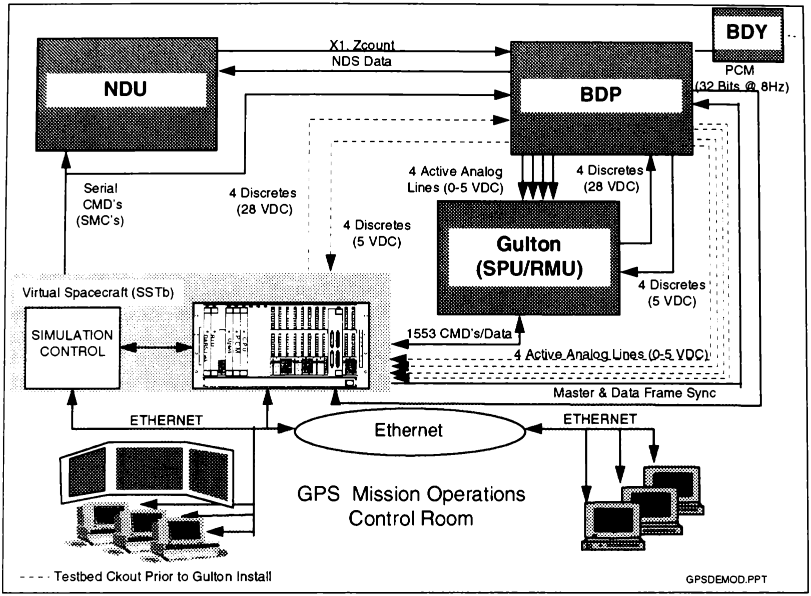


Figure 4 Subsystem Testing of The Nuclear Detection System/Burst Detection System Supports The Demonstration of End-To-End Interfaces; From The Control System Through To The On-Board Processor And Back To The User

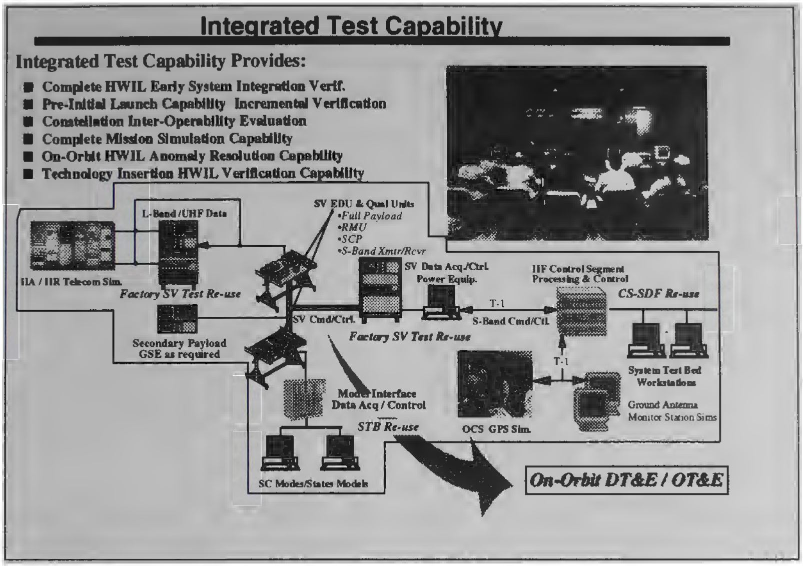
Future Activities

Existing secure MOSC communication links will be enhanced to support test connections to the Eastern Launch Site (ELS) and Falcon AFB. The GFE'd IIR SV Telecommunications Simulator and Autonav Emulator will be connected via Ethernet local area networks and/or hardline connections (Depends on whether STB's Telecom's capability moves to ELS or the Telecom's come to Rockwell, Seal Beach).

The laboratory continues to support satellite vehicle development and related subsystem definitions. Specific integrated test capabilities/functions, see Figure 5, that are in-work for the GPS program are as follows:

- Complete HWIL Early System Integration Verification
- Pre-initial Launch Capability Incremental Verification
- GPS Constellation Inter-Operability Evaluations
- Complete Mission Simulation Capability
- On-Orbit HWIL Anomaly Resolution Capability
- Technology Insertion of HWIL Verification Capability

These capabilities will support all GPS on-orbit Development, Test and Evaluations including Operational test and Evaluations.



Integrated Test Capability

Integrated Test Capability Provides:

- Complete HWIL Early System Integration Verif.
- Pre-Initial Launch Capability Incremental Verification
- Constellation Inter-Operability Evaluation
- Complete Mission Simulation Capability
- On-Orbit HWIL Anomaly Resolution Capability
- Technology Insertion HWIL Verification Capability

Figure 5 Integration of Our System Test Bed (STB) Factory Test Equipment, and Control System Provides A Total GPS System Process Capability

Acknowledgement

This work was accomplished under Rockwell's Space Systems Division's IR&D Project-301, Advanced Satellite Control Systems and Launch Vehicle Life Cycle Costs Concepts (October 1994 - March 1996). The authors wish to recognize the efforts of the following individuals who played key roles in this project: Liz Yingling, Will Wagner, Bob Ringquist, Jerry Simmons, Glenn Fox, Rick Gerardi, Russ Eigel, and Bob Schoen for the testbed and simulation infrastructure. Mike Herndon, Mik Mukae, Ed Nemeth, Huan Nguyen, George Salapski, George Wesselman, Leland Akasaki, Eric Clason, Claudia Ferguson, Larry Hung, Catherine McCaw, Peter Suk, and Dave Webb for the navigation data unit and the related navigation software. Special recognition is due to Larry Mah, Alberto Perez, and Willem Emmer for their test hardware support.

References

1. System/Simulation Infrastructure Programming Guidelines, Conventions & Configuration Management (S/SI PGCCM) 91930000D001D04-Rev.B (November 14 1995)
2. S/SI Programmers Reference Guide 91930000D002D04 (September 29, 1995)
3. S/SI Programmers Manual 91930000D001D05 (November 1995)
4. S/SI Users Manual 91930000D001D03 (November 1995)

AIAA-96-3537-CP

QUALIFYING AND TESTING HASI-STUB FOR HYUGENS-CASSINI MISSION

F. Angrilli[1], G. Bianchini[1], S. Debei[1], G. Fanti [2], B. Saggin[1]

1) Center Studies and Space Activities "G.Colombo"
c/o Department of Mechanical Engineering, University of Padova
via Venezia 1, 35131 Padova-ITALY tel. +39-49-827-6808- fax 049-827-6785

2) Department of Industrial Engineering, University of Parma,
Viale delle Scienze, 43100 Parma, Italy. tel. +39-521-90-5866- fax 0521-90-5705

Abstract

A new procedure for qualifying and testing sub-systems for space purposes is applied to the HASI-STUB for Huygens-Cassini Mission.

The procedure consists of developing structural, experimentally validated numerical models (EVNM), made up of macro-elements directly calibrated by means of laboratory tests.

Once calibrated, the models also easily allow the evaluation of system behavior to extreme space conditions not attainable on ground.

The mechanical and the thermal EVNMs of STUB were used to optimize the structure and to simulate some qualification and acceptance procedures and some extreme environmental conditions such as the thermal shock from 295 K to 77 K.

The high accuracy EVNM is a powerful tool that permits the validation of space hardware saving a lot of resources in terms of man power and money.

1. Introduction

In the Cassini Mission to the saturnian system, the Huygens probe has the major task to penetrate into the Titan atmosphere and land to the "surface" gathering as much as possible information on physical and electrical characteristics of Titan atmosphere and of its surface.

The environmental conditions, as known from the earth and space observations, during atmospheric descent are particularly critical and aggressive due mainly to the very low temperature (between 175 and 75 K) and the increasing pressure (of CH4 and other carbonium compound) up to around 1.5 bar at the surface, and lightning high probability.

The HASI experiment (Huygens Atmospheric Structure Instrument) is devoted to measure the relevant parameters of the Titan atmosphere during the entry and descent phase in order to reconstruct the temperature, and density profile. It's composed by four subsystems plus a DPU (Data Processing Unit) that process and combine the data into Probe telemetry format:

- STUB, that carries the temperature, pressure and acoustic sensors;
- ACC dedicated to measure the Probe accelerations during the entry and descent;
- PPI pressure profile instrument made up by the inlet pressure aperture (Kiel probe) attached to the Stub, and pressure sensors and electronics on a dedicated board inside the DPU;
- PWA permittivity wave analyzer and altimeter instrument which take advantage of 2 booms that will be deployed during the descent phase. [2]

In particular, the STUB is the HASI subsystem which will be exposed to the atmospheric environment of Titan (fig 1).

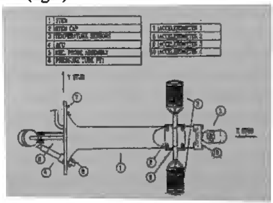


Figure (1)

It consists of a cylindrical structure in ergal with an ellipsoidal base (STEM) to interface with the Probe ring, which holds:

- a pair of temperature sensor (TEM) built on purpose by Rosemount Aerospace Inc.

of Burnsville (MN), each of them composed by one coarse and one fine sensitive element;

- an inlet pressure aperture (total pressure Pitot tube called Kiel Probe) optimized with respect to the drag forces, in stainless steel, attached to the Stem top, and the connecting pipe to take the flow inside DPU,

- an acoustic sensor fixed to Stem base, to detect natural acoustic phenomena of atmospheric events, based on elastic membrane with metallic strain gauges. Project requirements concern measurements of:

- atmospheric temperature and pressure profile in the range T = 70-300 K, with an accuracy of 0.1 K;

- acoustic pressure over a frequency range of 0-5 kHz, and static pressure in the range 0-2000 mbar with an accuracy of 5mbar. [1]

The qualification constraints of the Probe reflect on subsystems qualification in different ways; as for the STUB, the more demanding requirements come from:

- 1) very large thermal excursion(from 300 to 70 K) and thermal gradient since it will be exposed to the external temperature in few seconds when the Probe cover will be ejected

- 2) high structural stiffness and low mass to resist to launch acceleration, and to deceleration peak at parachute aperture. Stem has to hold in place without vibrations and deformations TEM,ACU and Kiel probe sensors

For this reason the qualification process of this part has stringent requirements (see next paragraph) in order to guarantee that it has been designed with adequate safety margins, maintaining its functionality and nominal performances.

The TEM sensors to measure the temperature of the Titan atmosphere during the descent consists of a Platinum-Rhodium support structure where the primary sensor is wounded while the secondary sensor is glued.

The primary sensor (fine) is made by a bifilar wire of Pt 99.999%, 0.1 mm of diameter and 2 m long insulated from the structure by a thin layer of glass. The platinum wire is coated by a parylene layer covered by gold coating.

The secondary sensor (coarse) is made by a thinner wire coil of pure platinum at 99.999%, (0.018 mm diameter) embedded in anniled glass, and glued on the front part of the structure. It has been designed for backup function with longer time constant but with a greather solidity.

2. Test philosophy

According to the standard procedures requested by space agencies, at the end of the Testing Procedures, the mathematical and numerical models of the subsystem, must be furnished

A detailed theoretical model of each unit is necessary to show that the subsystem can satisfy the design specifications. If the unit is composed of critical components, detailed models are necessary too.

A second, very rough, numerical model is requested for the unit integration in the probe model.

Following this standard procedure, such theoretical models have to be verified after the testing on the Structural, Engineering and Qualification Model

In the not unusual case in which the result is not in agreement with the test, the theoretical models must be improved.

In the case of HASI-STUB, a quite different approach was followed for building the thermo-mechanical numerical models; the highly accurate, so-called "Experimentally Validated Numerical Models" EVNM were built by means of an experimental calibration of the numerical model during designing, manufacturing and testing.

The EVNM consists of macro-elements named EFE "Equivalent Finite Elements" [4] capable to reduce the number of degrees of freedom of the mesh; the EVNM is also very accurate because each thermo-mechanical parameter and each kind of EFE are directly calibrated by comparison with physical units and critical elements.

While the standard procedure compares results a posteriori, the proposed procedure allows the direct calibration of the macro-elements with a better theoretical modellization of the component and the immediate possibility to improve the design if any critical aspect are found.

It is also possible to analyze effects of extreme space conditions and to evaluate parameters not simple to be measured such as the stress and the temperature distribution in a delicate instrument as TEM.

The mechanical and the thermal EVNM was built according to the procedure shown in figure 2.1. The STUB is the unit under investigation and the critical components are the STEM and the TEM because of its delicate structure and the required accuracy of ±0.1°C in the temperature measurements.

Initial EVNMs of the STUB, STEM and TEM were built from the analysis of the initial design, the thermo-structural theoretical behavior, the environmental conditions and the design specifications. During manufacturing and testing phase, the improved calibration of each EVNM was reached by a step-by-step technique;

a consequent improvement of the STUB and TEMs design occurred. From the information obtained by the EVNM, an optimization of the STEM structure was reached. The relatively high degree of accuracy obtained in the validated EVNM of STUB, STEM and TEM allowed an economical employ of such models as a powerful means in the analysis of:

- the hostile space ambient interaction with STUB;
- the simulation of functional tests in space environment,
- the validation of the theoretical behavior of each instrument;
- the evaluation of possible corrections of acquired data;
- the simulations of effects caused by severe qualification tests. The EVNM is also very useful because it can be used in substitution of expensive tests, this was done for the Shock Response Spectrum Test.

The STUB EVNM confirmed that ACU and KP sensors were not critical from a thermo-mechanical point of view. The critical aspects of the STEM are about a relatively reduced mass imposed by design specification; a relatively high stiffness in order to cause minimum accuracy errors in the sensors; a relatively low thermal conduction in order to dissipate the minimum thermal power.

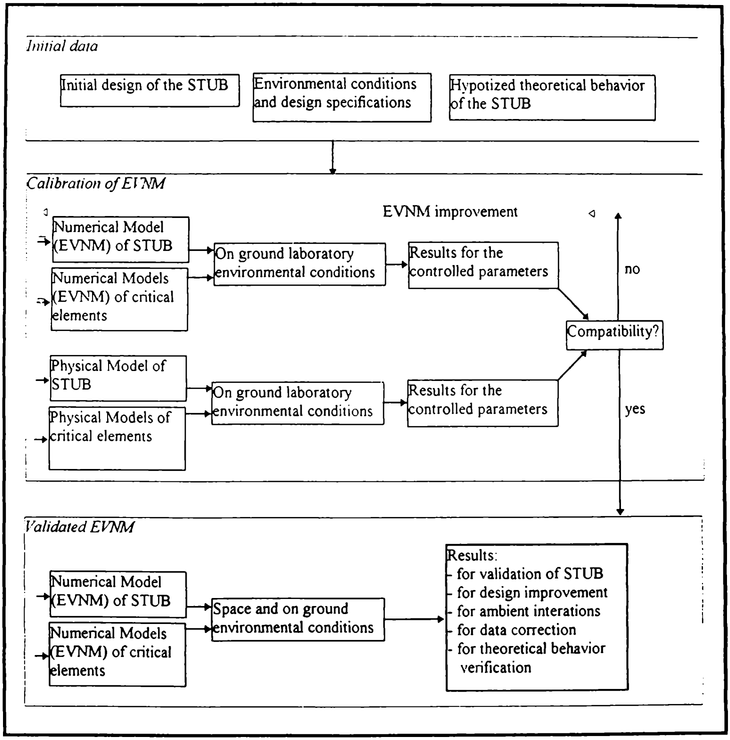

Initial data
Initial design of the STUB
Environmental conditions and design specifications
Hypotized theoretical behavior of the STUB
Calibration of EVNM
EVNM improvement
Numerical Model (EVNM) of STUB
Numerical Models (EVNM) of critical elements
On ground laboratory environmental conditions
Results for the controlled parameters
no
Compatibility?
Physical Model of STUB
Physical Models of critical elements
On ground laboratory environmental conditions
Results for the controlled parameters
yes
Validated EVNM
Numerical Model (EVNM) of STUB
Numerical Models (EVNM) of critical elements
Space and on ground environmental conditions
Results:
- for validation of STUB
- for design improvement
- for ambient interations
- for data correction
- for theoretical behavior verification

Figure 2.1: scheme of the proposed procedure to build the EVNM of the STUB.

3. Mechanical Model Calibration: tests verification e numerical computation

The calibration of the numerical model used for structural and dynamic purpose on the basis of experimental results has been performed. Taking into account static and dynamic equivalence between the numerical model and physical one, a FEM model has been developed. It consists of 418 nodes, 276 plates representing the STEM, 2 beams with concentrated mass, plus two plates representing each TEM, one beam and one concentrated mass plus 16 plates to represent KP, a concentrated mass for ACU. The attachment points to the probe were schematised with boundary elements, with axial stiffness, while the rotation relative to each bound has been left free.

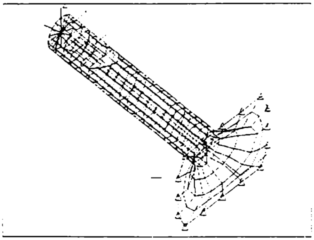


Fig. Mesh of HASI- STUB

The calibration of the numerical model was performed employing the Impulse Response Spectrum, which represents an accurate modal method to check the resonance frequencies. The STUB was constrained to an open box structure, with a mass of about 2.5 kg, made of aluminium, the same employed for qualification mechanical test, with some adjustment, bounded to a massive support. A resonance checking was performed on the interface, with a multipoint excitation, to avoid to excite only a particular mode. An accelerometer was positioned as close as possible to the bounding reference hole, obtaining a value of the first natural frequency of 1900 Hz.

The physical prototype of the STUB has been tested along the three principal axes: X longitudinal, along the axis of major M.o.M., Y transversal according the principal direction of the TEM, and Z.

The measuring accelerometers were positioned in four locations (see fig. 1):

- 1: near to the reference hole (to check the input excitation);
- 2: on the STEM CAP as close as possible to one attachment point of the TEM,
- 3: in proximity of the ACU,
- 4: in proximity of the KPA

By computing the Fourier spectrum of the input channel, and those related to the other channel of measurement, it was possible to detect the frequencies of resonance, in particular the first three, shown in the following table. These values were obtained with an uncertainty of +/- 2Hz, due to the spectral resolution of the acquisition system.

Tab. 3.1

Impulse	Computed	Swept Sine
244	246	257
291	294	310
342	344	340

On the basis of these experimental results, the calibration of the numerical model has been made, and the results obtained performing a numerical modal analysis, are reported in the second column.

Comparing the computed values with those obtained by experimental analysis, it was possible to interpret the modal forms. The first one (244 Hz), is correlated to torsion mode of the PT1 around X axis. It excites a flessional mode of the STEM in the xy plane. The second corresponds to a flessional mode of the PT1 on the XZ plane, show fig. 2. The third (342 Hz), is the first natural mode of the TEM, which bends along its principal axis (Y direction).

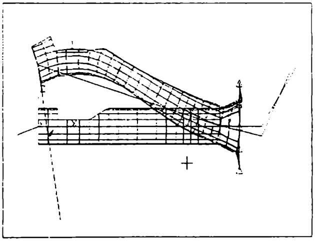


Fig.3.2 Third Natural Mode: 344 Hz.

Using th numerical validate model it was possible to evaluate, with accuracy, the higher frequencies of resonaces, that is:

- the fourth frequency at 356 Hz, which represents flessional mode of the STEM in xz plane;

- the fifth, at 965 Hz, which is fessional mode in the xy plane.

- the sixth, shown in fig 3.2. which flessional mode in xz plane

4. THERMAL MODELING OF STUB

The modeling of the STUB was intended to analyze two main problems:

-the disturbances effects on the TEM measurement due to the interference with the probe thermal field;

- the heat leak through the STUB, during the various mission phases in order to verify that it matches the project constraints.

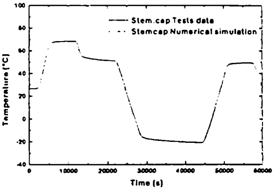
During the descent phase, the probe is subject to strong environmental changes, both in pressure and temperature. The main elements of the thermal configuration are the Titan atmosphere, a cold shield of the probe, holding the STUB, and thermally insulated from the experiment platform, the experiment platform, supporting the instruments electronics and the gas trapped inside. The numerical model had to include not only the STEM but also the pipe connecting the Kiel probe to the DPU, and the cabling between DPU and the sensors.

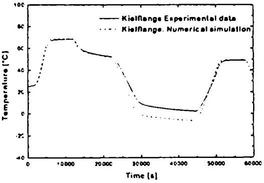
The numerical model has been built following the lumped parameter approach, required for using the ESATAN software. The whole system has been discretized with about 150 nodes coupled both radiatively and conductively; in addition three boundaries, whose temperature was provided by models and thermal analysis at higher level, i.e the Titan atmosphere, the Experimental Platform and the internal gas. The greatest uncertainties of the model are related to the "actual" environment that the probe will experience during its active phase, whose determination is the aim of the mission itself. However this unknown has been managed by the worst case analysis.

As a consequence the model has been checked in the vaccum, condition different from those foreseen during operational phase, to verify the correspondence between the behavior of the instrument and that of its model for the known part of the problem. The environment effect has been only analyzed by the parametric investigation. The selected test was therefore a temperature cycle under thermovacuum condition, foreseen also as qualification test.

The numerical model was therefore modified by removing the boundary nodes corresponding to the atmosphere and all the related convective links. What was expected as feed-back from this kind of test was the checking of the overall modeling procedure and of the uncertain parameters used in the numerical modeling, such as contact thermal resistances in mechanical joints, thermal resistances of the cables, emissivities of the surfaces.

Test conditions have been implemented in the numerical model, considering as boundary the surfaces of the vacuum chamber whose temperature was controlled and/or monitored during the test. The result is shown in figure () where the temperatures predicted by the numerical model are plotted along with those measured for two points of the stem. The discrepancies (about 4 K) in the behavior at Kiel probe have been assessed to depend from the mounting of the thermometer, directly exposed to the heat flux radiated from a hot screen.





Conclusions

The mathematical modeling procedure allowed to implement a numerical-experimental tool which was useful to foresee the response of the system to environmental stimulations. In particular the structural mathematical model allowed, using a reduced number of tuned elements to represent also high frequency vibrational modes, while the thermal model, validated in the thermo-vacuum test conditions, has been used to predict the instrument behavior in the complex environment expected during the mission.

References

1- ESA-SCI(88) CASSINI Saturn Orbiter and Titan Probe

2- M.Fulchignoni, F.Angrilli, G.Bianchini . et al "The Atmospheric Structure Instrument (ASI)" proposal for ESA Huygens Probe Mission-Rome 1989

3- ESA EID part A Dec.1992 "Huygens Probe-Cassini Mission Experiment Interface Document"

4. F. Angrilli, G. A. Bianchini, G. Fanti
Dynamic Analysis of a Pierced Plate Under Impulsive Loading.
Proceedings of "International Conference on Numerical Methods for Transient and Coupled Problems", pp 275-285, Venice, Jul.1984.

5 F. Reccanello, G. Fanti, F. Angrilli
Boom Structure Optimization for HASI on Cassini Mission
AIAA/ASME/ASCE/AHS/ASC 35th Structures, Structural Dynamics and Materials
Conference, Hilton Head, South Carolina, U.S.A., April. 18-20, 1994.

6 F. Reccanello, F. Angrilli, G. Bianchini, G. Fanti
Testing and Optimization of a Structure for pressure and Temperature MeAsurements in Planetary Atmosphere
36th AIAA/ASME/ASCE/AHS/ASC Structure, Structural Dynamics and Material
Conference, New Orleans, LA, U.S.A, April 1995.

AIAA-96-3538-CP

IMPROVING MUNITION SIMULATION FIDELITY THROUGH USE OF AN ORDNANCE SERVER

Dicola, John; Fischer, Peter; Mutschler, David; Ullom, Lawrence[*]
ACETEF/MFS, Naval Air Warfare Center - Aircraft Division Patuxent River, MD[†]

Abstract

A recurring problem in Distributed Interactive Simulation (DIS) exercises is the use of different munition models by different entities in the same exercise. Even when these entities simulate the same weapon, the use of different munition models with their varying levels of fidelity can unjustly favor the performance of one combat platform over another. Since validated weapon simulations with realistic flyouts and trajectories may perform differently than less complicated models, the accuracy of the simulation can suffer greatly. This paper describes a solution wherein munition simulations are separated from launching platform simulations and incorporated into a single **ordnance server**. This server provides multiple entities in an exercise with a uniform set of munition models. When a munition is launched, the server utilizes the best available model to ensure the most correct simulation of the munition and controls it appropriately. Thereby, overall simulation fidelity is improved. In addition, the ordnance server facilitates simulation design and management since munition models may be developed, tested, and verified independently of the simulation of any launch platform.

Introduction

Distributed Interactive Simulation (DIS) was designed from the beginning to bring a wide variety of different simulators and simulations together in a unified synthetic world. This primary goal leads to a recurring inherent problem with the use of different munitions by different entities in the same exercise. The use of different munition models with their varying levels of fidelity can unjustly favor the performance of one combat platform over another. This raises the issue of a "fair fight" when using data from these exercises to make aquisition decisions, evaluate measures of effectiveness in training, or develop strategic doctrine.

Due to the different requirements of each of these groups, it is not possible to pick a standard model and distribute it for inclusion in each simulator in an exercise. This paper describes a better solution wherein munition simulations are separated from launching platform simulations and incorporated into a server. This approach can increase the overall fidelity of an exercise. It also aids configuration management by separating the weapon models from the launch vehicle.

The ordnance server can provide a level playing field by incorperating models of equal fidelity for all weapons used in an exercise. Munition modeling is no longer driver by the requirements of particular simulators. Moreover, the exact level of fidelity can be driven by the particular exercise requirements. The server takes over the modeling of weapons in an exercise from the initial fire event generated by the launching entity and controls the munition through it's termination.

The ordnance server concept was originally used with man in the loop flight simulators to support classified and unclassified munition modeling. The method was later tested with synthetic entities in a large scale training exercise and proved viable. Additional research is needed to determine efficient mechanisms for passing prelaunch and inflight data from the launch platform to the munition model. With contined development the

[*] Any correspondence can be sent to Commander, Attn: Alexandra Wachter, Naval Air Warfare Center Aircraft Division, 48140 Standley Road, Patuxent River MD 20670-5304 or via e-mail at wachteram%am6@mr.nawcad.navy.mil.

[†] Mr. Fischer is employed at J.F. Taylor, Inc. All work was conducted at the Manned Flight Simulator, NAWCAD Patuxent River, MD.

ordnance server can greatly improve the valididty of results obtained from distributed simulation exercises.

Standard DIS Missile Simulation

Three types of PDUs are used to describe a tracked munition simulation in a DIS exercise: Fire, Entity State, and Detonation. The Fire PDU is issued by the firing entity as an initiation of the munition. It contains information including the identity of the firing entity, the identity of the intended target, the type of munition fired, the initial velocity of the munition, and the location from which the munition was launched. Thereafter, Entity State PDUs are issued to describe the resulting trajectory of the munition. A Detonation PDU is issued at the end of the munition flight. It communicates the location of the burst (if there was one), the detonation result, and other related information. It is the target simulation's responsibility to assess the damage to the target from the information provided in the Detonation PDU.

Missile Simulation by Ordnance Server

Normally, these PDUs are issued by the firing entity's simulation application. However, when the Ordnance Server is used, the firing entity's simulation application only issues the Fire PDU. The Ordnance Server is responsible for using the information from the Fire PDU, and other ground truth information available over the DIS network, to properly issue the Entity State and Detonation PDUs for that munition.

This methodology requires the same amount and type of PDU traffic as that required for a standard entity simulated munition launch. The network sees no additional load due to the utilization of the Ordnance Server.

Determination of Pertinent Fire PDUs

Before the exercise begins, the Ordnance Server is configured to simulate missiles for all entities from a particular site, application, or entity. Upon receiving a Fire PDU from one of these entities, the Ordnance Server determines if the type of missile fired is one that it has been configured to simulate. If so, the missile simulation is initiated. Otherwise, the Fire PDU is ignored. This configurable filtering gives the user a great deal of flexibility as to the extent to which the Ordnance Server is used, versus the extent to which an application simulates its own munitions. Also, this flexibility can be used to further distribute the processing load. One Ordnance Server can be configured to simulate all munitions from one application, while another Ordnance Server is configured to serve another. In another case, one Ordnance Server could be configured to simulate all guided munitions, while another one simulates the ballistic munitions. This flexibility is important due to the highly variant nature of the scopes and requirements for different DIS exercises.

Basic Functionality of Ordnance Server

All configuration and control of the OS can be accomplished, by the user, by way of the OS's Graphical User Interface (GUI). The OS was designed to be maximally configurable. Beyond the configurable filter criteria, described earlier, the OS's GUI can be used to

- select the terrain database to use for ground impact checks,
- provide the mapping of missile type input (via Fire PDU) to munition model used,
- freeze the munition simulation(s) and then resume them,
- input the guidance method and aerodynamic behavior of generic munition models,
- describe the radar emissions produced by actively guided munitions,
- select the fuse and warhead types of the munitions,
- select the type of runtime feedback to be provided to the user,
- and input other detailed simulation information.

The OS's GUI provides configuration data to the OS Executive. The OS Executive, in return, provides runtime feedback about munition and OS performance to the GUI for display. OS configuration data can be saved in and loaded from configuration files to ease the initialization process during subsequent exercises.

The OS includes a set of built-in generic munition models with configurable guidance methods and aerodynamic behavior. When a Fire PDU is received via the DIS Gateway, the OS's PDU Engine routes the pertinent information to the OS Executive. If the Fire PDU is from a client entity of the OS and it's munition type is mapped to a generic missile model, the OS Executive then initiates the proper built-in generic fly-out with the selected guidance model. The built-in fly-out model keeps track of the current state of the munition model by way of a munition database that it continuously updates and accesses during the munition's

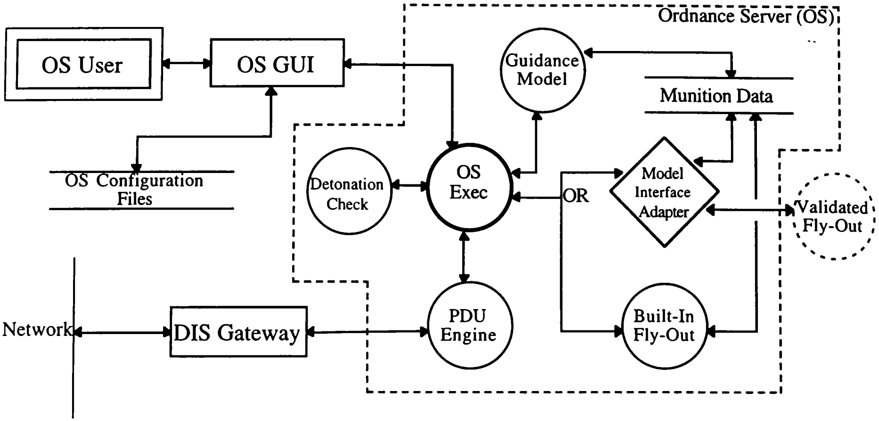


Figure 1: Simplified Block Diagram of Ordnance Server

flight. The OS can simulate multiple concurrent munition fly-outs by simply adding new munition records to this munition database and then updating them in turn. Also during the munition flight, the built-in fly-out provides current data about the munition's position, velocity, and orientation to the OS Executive. The OS Executive routes this information the PDU Engine for creation of the munition's Entity State PDUs. The PDUs are then broadcast over the network by the DIS Gateway.

During a guided munition's flight, the DIS Gateway keeps track of external entities and emissions that could affect the guidance of the munition. The OS Executive continuously provides this information to the guidance model so that it can properly model the input to the missiles sensor(s) and guide the munition's fly-out model in a realistic manner.

Regularly, during all munition fly-outs, detonation checks are made. The criteria that causes a generic munition to detonate is highly dependent on the fuse type specified by the user at configuration time. The OS's generic munitions can be altitude, depth, proximity, contact, or time fused. Detonations can be caused by external detonations, collisions, ground impacts, or entity impacts. Information about these external entities and events is provided by the DIS Gateway. When the munition's detonation criteria has been met, the OS Executive sends the detonation information to the PDU Engine for creation of a Detonation PDU, this PDU is then broadcast over the network by the DIS Gateway. The OS executive also deactivates the built-in fly-out model. Upon deactivation, the fly-out model removes the munition's data from the munition database so that no more Entity State PDUs will be issued for that munition.

Interface Between OS and Validated Munition Models

When a level of fidelity beyond that of the generic missile models is required, the Ordnance Server provides the use of validated munition models. These models are integrated into the Ordnance Server by developing an interface between the existing validated model and the existing OS code. This interface is called the Model Interface Adapter.

Functional flow within the OS during an interfaced external munition simulation is identical to that of a built-in fly-out model simulation (described earlier) with the following exception: the functionality of the built-in fly-out model is replaced by the union of the Model Interface Adapter and the interfaced validated fly-out model. The functions of the Model Interface Adapter are

- to make the external munition simulation "look like" (as far as the type and quantity of data being passed in and out) the OS's built-in fly-out model to the OS executive and

- to make the OS "look like" the simulation executive, under which the external model was designed to run, to the external fly-out model.

Most of the external munition models that the OS currently supports were originally developed for the Tactical Aircrew Combat Training System (TACTS). TACTS munition models have undergone a fairly extensive validation and verification process and are very good representations of the missiles they emulate. The TACTS missile simulations were also good candidates for integration into the OS because they are easily hosted on a SGI, have a well defined interface, are capable of being called cyclically, and run in real time.

Some differences between TACTS models and the built-in OS models are location and orientation representations, required cycle periods, and the variety of runtime feedback produced. All of these differences were resolved by the Model Interface Adapter. For example, TACTS missiles provide extra runtime feedback in the form of descriptive missile failure messages and probability of kill estimates. To ensure that this information is not wasted, additional functionality was added to the OS's GUI so that this data is provided to the user as needed. The probability of kill estimates are used to determine the detonation result reported in the munition's Detonation PDU.

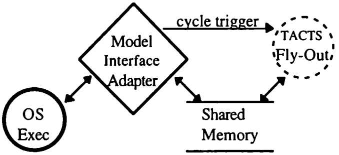


Figure 2: OS/TACTS Interface

The TACTS models were designed to interface with their simulation executive through shared memory blocks. These are accessed by the Model Interface Adapter by mapping equivalently sized data structures over the shared memory locations maintained by the TACTS models. The TACTS model's cycle time is controlled by the Model Interface Adapter by calling a parameterless TACTS routine which causes the TACTS model to read and update the shared memory interface.

Available Munition Models

The TACTS munition models listed in the following table have been integrated into the OS. Additionally a Tomahawk Trajectory Generation Tool, developed by the Dahlgren Division of the Naval Surface Warfare Center, has been successfully interfaced by the OS. Additional TACTS and other types of validated munition simulations will be added to the OS to ensure that the OS can provide the best available model needed by prospective client applications.

TACTS Models	Common Name
AA-2B, 2D	Atoll
AA-8A, 8C	Aphid
AA-10A, 10B, 10C	Alamo
AA-11	Archer
AIM-7F, 7M, 7E2	Sparrow
AIM-9G, 9H, 9L, 9M, 9P3	Sidewinder
AIM-54A, 54C	Pheonix
AIM-120	AMRAAM
FIM-92	Stinger
R-550	Magic
SA-2	Guideline
SA-3	Goa
SA-4	Ganef
SA-5	Gammon
SA-6	Gainful
SA-7	Grail
SA-8	Gecko
SA-9	Gaskin
SA-16	
SA-N-3	Goblet

Table 1: TACTS Models
Currently Integrated into the Ordnance Server

Simulation Design and Management

Since the Ordnance Server runs independently of the simulation of any launch platform, munition models can be developed and integrated without having to account for internal differences between the various OS client applications that will be launching them. The programming language, structure, and ideology of the client applications are all completely transparent to the Ordnance Server. Any development of new or old munition simulations on the Ordnance Server automatically enhances the munition simulations of all client applications that use them.

The structure and content of Fire PDUs is well defined by the DIS standard. Since the Fire PDU is used as the interface between the Ordnance Server and firing client applications, an Ordnance Server tester can assume that if a model performs correctly when launched by one application that issues properly encoded Fire PDUs, it will work with all applications that issue properly encoded Fire PDUs. For instance, all testing of a Sidewinder launched from a DIS compliant F-18 simulation need not be duplicated for a Sidewinder launched from a separate, DIS compliant F-14 simulation.

Verification of the munition models is also facilitated by the use of an Ordnance Server. If munition models are modeled by the firing entity's simulation applications, the determination that a missile correctly approximates the actual behavior of it's real counterpart must be made for each application that uses the missile. Even then, there is no guarantee that duplicate missile types will behave identically from application to application. With the use of the Ordnance Server, on the other hand, verification needs to be completed only once for each of the types of munitions modeled. Since all simulations share the exact same set of munition models, consistent missile behavior is assured.

Conclusion

The Ordnance Server provides an efficient, logical means of separating munition simulations from simulated launching platforms. As a stand-alone munition simulator that initiates weapon launches via Fire PDUs, the OS provides a non-intrusive method for multiple simulations to share a uniform set of weapon models. It allows independently validated, high fidelity munition models to be easily integrated into a simulation or set of simulations. Sharing munition models leads to more realistic exercises and moves modeling and simulation closer to providing a "fair fight" by eliminating weapon fidelity variations.

Acknowledgements

The authors wish to thank the following people for their assistance in developing the Ordnance Server: Paul Maassel for concept exploration and refinement, Rob Kerr and Kimberly Neff for implementation, and Al Gramp and John Phillips for their support in integrating the TACTS models. Lastly, the authors would to thank Alexandra Wachter for organizing and leading this team and for providing the spark and support necessary for this paper's creation.

Introducing Parks College Reconfigurable Engineering Simulator

Lawrence G. Boyer[*]
Parks College of Saint Louis University
Cahokia, IL 62206

ABSTRACT

The Aerospace & Mechanical Engineering Department at Parks College has obtained a Reconfigurable Engineering Flight Simulator. The achievement of this goal was due partly to McDonnell Douglas Foundation having awarded Parks College a grant in 1994 to be used for the procurement of a simulator. The resulting simulator, however, is estimated to be worth some three times the MDF grant. The additional funding came from OPINICUS Corporation of Tampa, FL who designed, integrated and manufactured the simulator.

The purpose of the subject simulator is to enhance the educational capabilities of the undergraduate and graduate programs and provide a facility for simulation studies and research. This paper describes the newly acquired Parks College Reconfigurable Engineering Simulator and how the intended teaching and research usage led to the design specification.

INTRODUCTION

The design and evaluation of Aerospace Vehicle concepts is the focus of the engineering curriculum at Parks College. Design evaluation is a key element of the design exercise consisting of bringing design parameters into a known acceptable range. The simulator can be used to evaluate a specific design as well as demonstrate general flight vehicle design parameters and concepts.

PROCURING THE SIMULATOR

The procurement of an engineering simulator for Parks College was initiated with a grant from the McDonnell Douglas Foundation. By the time MDF committed to the grant, Parks faculty knew of other educational engineering simulators in use at other colleges, universities and military academies. The 'bottom line' (level of funding) being predetermined, a decision had to be made whether to procure two or three small desk-top simulators, or one high-fidelity unit. Early in the discussions the latter won out. The key argument was that smaller simulators could be procured or constructed at a later time but obtaining another major grant for a high-fidelity simulator could be difficult. The resulting Request For Proposal (as it was termed) was to a certain extent a "wish" list, but it would have been unproductive to include obviously unaffordable items such as a motion base. A single seat, open cockpit arrangement was judged adequate and so there was no request for a dual seat / dual controlled option.

Since Parks has a heritage as an 'air" college, manned air vehicles were the emphasis, but it was also desired that the simulator be reconfigurable enough to allow simulation of unmanned and non-air vehicles. Diversity of vehicle simulation, then, was a requirement. This meant that replication of a particular cockpit was not desired. A Flight Training Device was not appropriate. Neither were ancillary systems part of the required simulation. Therefore, a generic and reconfigurable cockpit was the answer.

Who will use it and how?

The desired device would provide high fidelity simulation of a vehicles' dynamic loop; from the control input, through the vehicle response, to the feedback experienced by the pilot. There are many ways such a device could be utilized in an engineering education environment.

One of the primary purposes envisioned for this simulator was for faculty or graduate students to conduct research on a wide variety of airframe, engine and control configurations. This would include such diverse areas as high lift device effects; autopilot; stability augmentation and control law design (including hardware in the loop); conventional and V/STOL aircraft and helicopter flying qualities research.

Other anticipated uses for this simulator included conducting 'flight testing' and gathering real-time data,

[*] Assistant Professor of Aero & Mech Engineering, Member AIAA.

teaching the art of real-time simulation and enhancing graduate and undergraduate stability and control courses.

Cockpit Hardware

The hardware components observable by the vehicle pilot were considered as these five:

- Controls, throttles & switches - what the pilot manipulates
- Instruments - what the pilot observes within cockpit
- Visual - what the pilot sees out the window
- Motion - what the pilot feels overall
- Sound - what the pilot hears

Early on it was decided that 'control loading' and the best affordable 'visual' were musts! It was also recognized that these two items would represent a significant portion of the simulator cost. Motion was definitely out of our price range.

Software considerations

Since software can be rather nebulous, our requirements in this area were much more difficult to specify and obtain.

For the undergraduate student user - a turnkey system

Ease of operation and modeling would be needed to allow development of new aircraft models rapidly and easily from known coefficient data without the need for programming skills. A highly user friendly menu driven interface would be essential for students to use the simulator to enhance undergraduate course work with minimum preparation.

For graduate student or faculty projects

Advanced users (with proper authorization) would need the flexibility of writing their own FORTRAN or C programs which could be added to or replace existing modules in a simulator load.

For the System Administrator - unlimited access

To provide maximum flexibility and full ownership, all software would need to be fully accessible to Parks, including source code, executives and inter-module data definitions. Such an open software architecture would not prevent adding, deleting or modifying modules or rearranging their execution sequence.

A Tall Order?

A system which is turnkey to the undergraduate but open architecture to experienced faculty would require multiple levels of access. Three were specified. Student (most limited), faculty (some limitations) and system administrator (unlimited). Further, full access to all the source code in a device like this is rare. Simulator companies make part of their living from software upgrades. But being a non-profit academic organization, the faculty hoped to find a company which was willing to accommodate our desire for full access to all the source code with the express understanding that all software stays at Parks.

To our knowledge, there were no off-the-shelf simulators fitting our requirements. A specification for the simulator was written and sent out as a Request for Proposal to select companies that were thought to be appropriate to the manufacture of this class of simulator. The RFP resulted in four proposals, each of which had some attractive features. The selected proposal by OPINICUS Corporation, however, essentially met all of the RFP requirements. The simulator was delivered in May of 1996 (see Figure 1). Most of the original text for the 14 hardware items and 23 software items in the RFP are presented later on as a catalog of the simulator's features and capabilities.

SIMULATOR FAMILIARIZATION

Having a few weeks to explore the capabilities of the simulator at the time of this writing, a few of the more outstanding features of the simulator can be presented.

General Reconfigurability

Cockpit configuration options include helicopter, fighter, transport and general aviation (tricycle and tail dragger). The pitch and roll control can be either a center stick (fighter, helicopter, aerobatic), a yoke/column (transport, general aviation) or a side force stick. The first two are loaded.

Three software models were delivered (fighter, helo and heavy). In time, the number of models will increase, limited only by disk space (1 gigabyte). Non-flight vehicles are also a future possibility.

Source Code Accessibility

It's hard to put a price on the ability to change virtually anything in the software. For Parks College this accessibility greatly increases the utility and life of the simulator as well as allows it to be used by faculty and students in other departments such as the Computer Science program.

Auto Test

This is a feature usually found only on full flight simulators and is a valuable, time-saver for demonstrating typical dynamic modes such as dutch roll and phugoid. Virtually any scenario can be programmed in the auto-test script file format.

Control Loading

The Fokker Electric Control Loading System vastly increases the utility of the simulator. Practically any control system can be simulated whose mass and inertia exceed that of the ECL system. The center stick could be modeled as a rigid or semi-rigid force stick as well. Simple harmonic motion demonstrations could be setup by using the ECL stand-alone.

IOS

The IOS (by ICONIX Corporation, Tampa, FL) is as user-friendly and easy to program as it is functional and attractive. Just a touch of a button invokes the editor and allows modification of any page. Pages are programmed in an intuitive "C like" language.

Changing aerodynamic coefficients is done on specially formatted pages which show each of the six coefficients in equation form (see Figure 2). Clicking on a particular term brings up an edit session where the user may make desired changes.

Parameter plotting is impressively straight-forward with many user options for plot presentation. Real-time data gathered during a test is displayed during the maneuver. The resulting plot can be reformatted, auto-scaled or custom-scaled and then plotted to the laser printer. Data tables can be plotted for inspection and graphically edited by clicking and dragging datapoints.

Visual

The **Ball 944** image generator with **Electrohome** projector greatly adds to the impression of realism. Without a good visual the simulator would not make much of a show to visitors, board members and administrators. For serious stability and control demonstrations to students, running a dynamic mode maneuver and observing the resulting motion from outside the aircraft enhances conceptual understanding more than hours of chalkboard explanation.

REQUEST FOR PROPOSAL EXCERPTS

Hardware

1. Single seat full size cockpit.
2. Simulated instrument suite using high resolution color graphics (see Figure 3).
3. Throttle quadrant.
4. Gear selector switch.
5. Flap selector lever.
6. Speedbrake selector lever.
7. Stability augmentation switches on programmable touch screen.
8. Autopilot annunciator lights and switches on programmable color touch screen.
9. Primary flight controls.
   a) Three channel electric control loading system.
      1) Center stick.
      2) Yoke and column through floor.
      3) Center stick cyclic and twist grip on collective (helicopter).

   Note: All of the above include actively loaded rudder pedals with integral spring loaded toe brakes.

   b) Force-operated sidestick (fly-by-wire).
10. Trim switches and trim position indicators for each of three axes on programmable touch screen.
11. I/O system expandable to handle future 'hardware in the loop' stimulation.
12. Simulator control and parameter display console for one-man operation.
13. Visual system.
14. Sound (aural cue) system.

Software

1. Highly user friendly menu driven user interface for simulator preparation, operation, model changes, data recording and plotting, test input setup and execution. This is essential for students to use simulator with minimum preparation.

2. All software fully accessible to customer, including source, executives and inter-module data definitions. Contractor protected by non-disclosure

agreement. All software for Parks College use only. Software architecture will not prevent adding, deleting or modifying modules or rearranging their execution sequence. The delivered load will be left unchanged and will serve as a benchmark.

3. Multiple software load configurations.

4. FORTRAN and C source code supported. A given load could be comprised of some FORTRAN source modules and some C source modules.

5. Simulator software architecture and hardware capacity must be able to accommodate a graphical control law modeling tool of our choice (such as MATLAB, XANALOG, etc.) which generates output in C or FORTRAN source code.

6. Full six degree of freedom, fully cross-coupled, non-linear, rigid body equations of motion. Fully aerobatic - no singularities.

7. Real-time executive. A multi-tasking dynamic pre-emptive scheduler. Modules critical to minimizing transport delay must run at the basic iteration rate (60 Hz). This rate but can be changed to 30 or 90 Hz. Other modules, such as the engines, run at a lower rate. Transport delay less than 150 msec from control input to end of first visual update is desired when load runs at 30 Hz. basic iteration rate. A method of verification must be provided.

8. Classical coefficient based aero model. For any coefficient the user has the option of a fixed coefficient or a table lookup to represent nonlinear aerodynamic data over entire flight envelope.

9. ASCII format lookup tables - up to four independent variables. The software will provide the flexibility for the user to add table lookup functions to any module.

10. International Standard Atmosphere simulated.

11. Weight cg and inertia simulation options.
    a) Fixed selectable values.
    b) Programmable time varying model.

12. Ground reactions simulation.
    a) Tricycle gear with steerable nose wheel through rudder pedals.
    b) Castoring effect feeds back through control loading.
    c) Optional tail wheel configuration.
    d) Optional helicopter skids.
    e) Software provides flexibility to completely redesign and reprogram ground reactions model.
    f) Selectable field elevation (visual is referenced to this same value).
    g) Effect of aircraft carrier deck motion included.

13. Reconfigurable simulated instrument suite.

14. Operators console.
    a) Display pages completely reprogrammable.
    b) Altitude, position, weight, cg, airspeed and heading selectable individually at any time.
    c) Initial condition sets scripted or snapshot.
    d) Freezes will include:
        - TOTAL FREEZE
        - FLIGHT FREEZE
        - ALTITUDE FREEZE (rate of climb includes climb acceleration factor)
    e) Failure modes (malfunction) pages (see below).

15. Failure modes (malfunctions). Simulator will include failures selectable through Operators Console. Software architecture will allow easy modification and addition of failures.

16. Navigation model.
    a) Up to 25 stations.
    b) ILS, MKB, NDB, VOR, DME.

17. Real time data record.
    a) Time stamped with user selectable time offset.
    b) Records user selected parameters to disk file.
    c) Disk file can be downloaded for use in other analysis packages.
    d) Maximum of fifty parameters per sample.
    e) Sample rate selectable.
    f) All parameters in load accessible - global as well as local data.
    g) Handles reals, integers and booleans.

18. Graphical data plotting.
    a) Plots to color graphics monitor while recording (if option selected)and when retrieved from disk.
    b) Scales and format selectable.
    c) Optional auto-scale.
    d) Hardcopy option.

e) Save on disk option.
f) Handles reals, integers and booleans.
g) Recorded data can be displayed in tabular format, or as x-y plot of selected parameter pairs, or in strip chart format.

19. Parameter Stimulate feature. Allows programming of a control input and command of its execution.
    a) Value taken as absolute or relative depending on selection.
    b) Capable of selectable time offset.
    c) Up to twenty simultaneously commanded parameters.
    d) Handles reals, integers and booleans.
    e) Hardware input individually bypassed in this mode.

20. Symbolic Parameter Look utility.
    a) Can display, monitor, change, etc. local or global data.
    b) Normally interactive display.
    c) Optional output to printer.
    d) Accepts either keyboard input or script file input.
    e) Utility can be invoked by another program such as the Automated Flight Test Sequencer (see below) in a non-interactive mode.

21. Automated Flight Test. Facilitates initializing, execution and recording of flight test maneuvers, both static trims and time histories via preprogrammed script files without a human pilot actually flying the simulator.
    a) Interactive Sequencer.
        1) Accepts test selection.
        2) Causes the Symbolic Parameter Look utility to read appropriate script file where the particular initial conditions for selected test are stored.
        3) Displays test status to user.
        4) Sequences through static trims using Automatic Trim Modes (see below) and as directed in script file.
        5) Activates time histories using Parameter Stimulate feature and as directed in script file which will also contain predetermined control inputs/actions.
        6) Activates printout and/or plot of results as directed in script file.
        7) Goes on to next test if any.
        8) Manual fly-out option selectable.
    b) Entire architecture reprogrammable.

22. Automatic Trim Modes. For specified flight condition, trim mode and trim parameters - aircraft, engines and controls trim automatically.
    a) Trim modes include:
        1) Longitudinal trim.
        2) Steady heading sideslip.
        3) Coordinated turn at constant airspeed.
    b) Can create more modes and/or modify delivered modes.
    c) Trim modes can be activated through Operator's Console or through Automated Flight Test.

23. Control Loading simulation.
    a) Programmed in a high level language such as C or FORTRAN.
    b) Includes aerodynamic surface hinge moments as fixed coefficients or nonlinear table lookups.
    c) Interface with host clearly defined and reconfigurable.
    d) Autopilot / stability augmentation servo motors simulated on all three axes.
    e) Steerable nosewheel/tailwheel with force feedback to pedals.
    f) Capable of simulating artificial feel.

24. Visual simulation.
    a) Generic airport scene including runway.
    b) Aircraft carrier scene - carrier state vector driven from host computer.
    c) Air-to-air target driven from host according to flight pattern selected by user through Operator's Console.
    d) Visual view selection. This feature allows the selection of either the normal out the window view or a side, rear or top view of own ship in order to observe it's motions.
    e) Reprogrammable simulated HUD.
    f) User may program additional visual scenes.

CONCLUSION

The Reconfigurable Engineering Simulator is a fresh and attractive addition to the suite of educational and research facilities at Parks College. There is a lot of work to be done in integrating the simulator into existing courses. This may be the subject of a future paper.

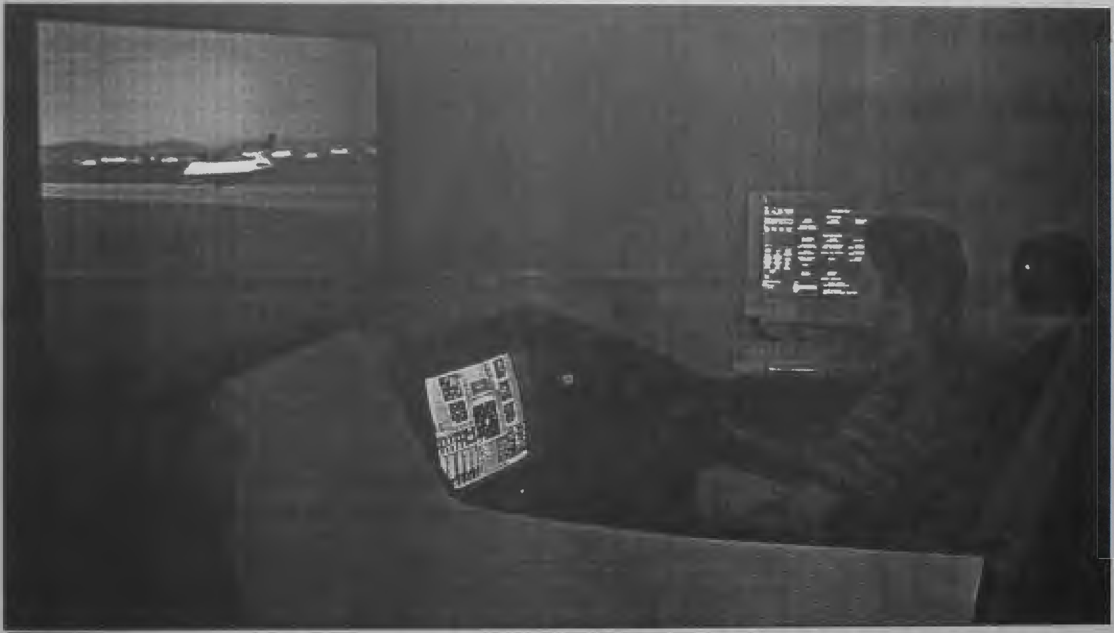


Figure 1. Parks College Reconfigurable Engineering Simulator built by OPINICUS Corporation.

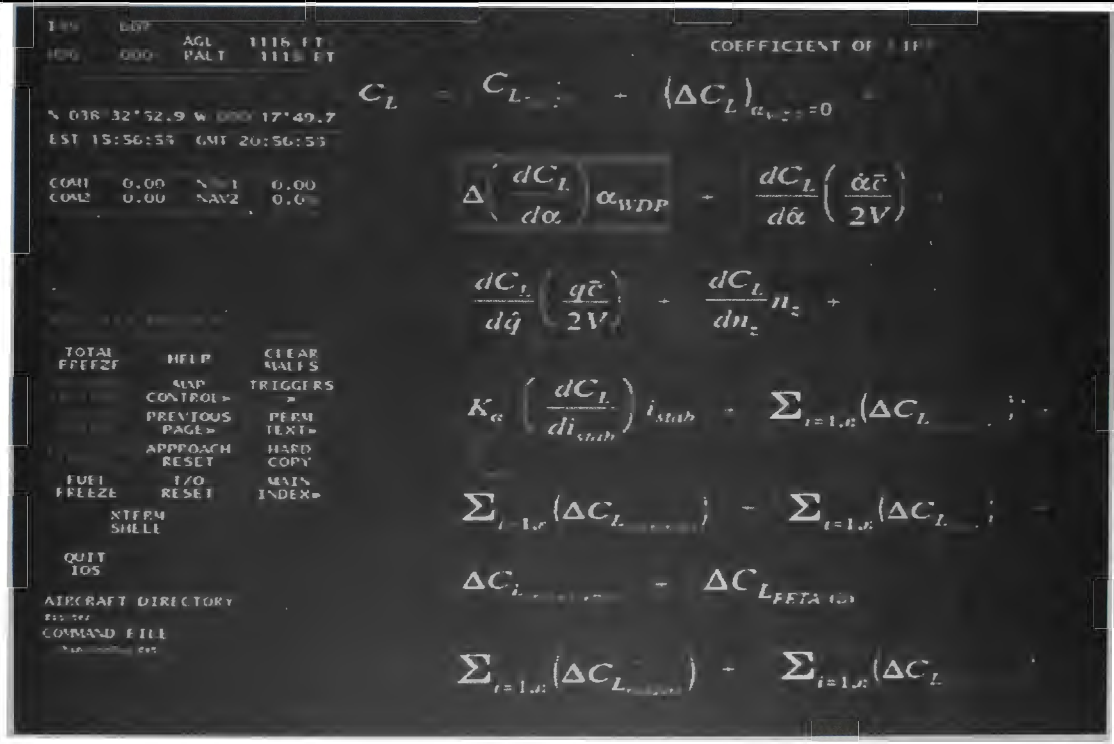

AGL 1115 FT
PALT 1115 FT
COEFFICIENT OF LIFT
N 036 32'52.9 W 000 17'49.7
EST 15:56:53 GMT 20:56:53
COM1 0.00
COM2 0.00
NAV2 0.0
TOTAL FREEZE
HELP
CLEAR MALFS
MAP CONTROL▸
TRIGGERS ▸
PREVIOUS PAGE▸
PERM TEXT▸
APPROACH RESET
HARD COPY
FUEL FREEZE
T/O RESET
MAIN INDEX▸
XTERM SHELL
QUIT IOS
AIRCRAFT DIRECTORY
COMMAND FILE
$C_L = C_L + (\Delta C_L)_{\alpha=0}$
$\Delta\left(\frac{dC_L}{d\alpha}\right)\alpha_{WDP} + \frac{dC_L}{d\dot{\alpha}}\left(\frac{\dot{\alpha}\bar{c}}{2V}\right)$
$\frac{dC_L}{d\hat{q}}\left(\frac{q\bar{c}}{2V}\right) + \frac{dC_L}{dn_z}n_z +$
$K_\alpha\left(\frac{dC_L}{di_{stab}}\right)i_{stab} + \sum_{i=1,n}(\Delta C_L$
$\sum_{i=1,n}(\Delta C_L) + \sum_{i=1,n}(\Delta C_L$
$\Delta C_L + \Delta C_{L_{FETA}}$
$\sum_{i=1,n}(\Delta C_L) + \sum_{i=1,n}(\Delta C_L$

Figure 2. Sample IOS page showing Lift Coefficient equation and contributing terms. Each term is selectable.

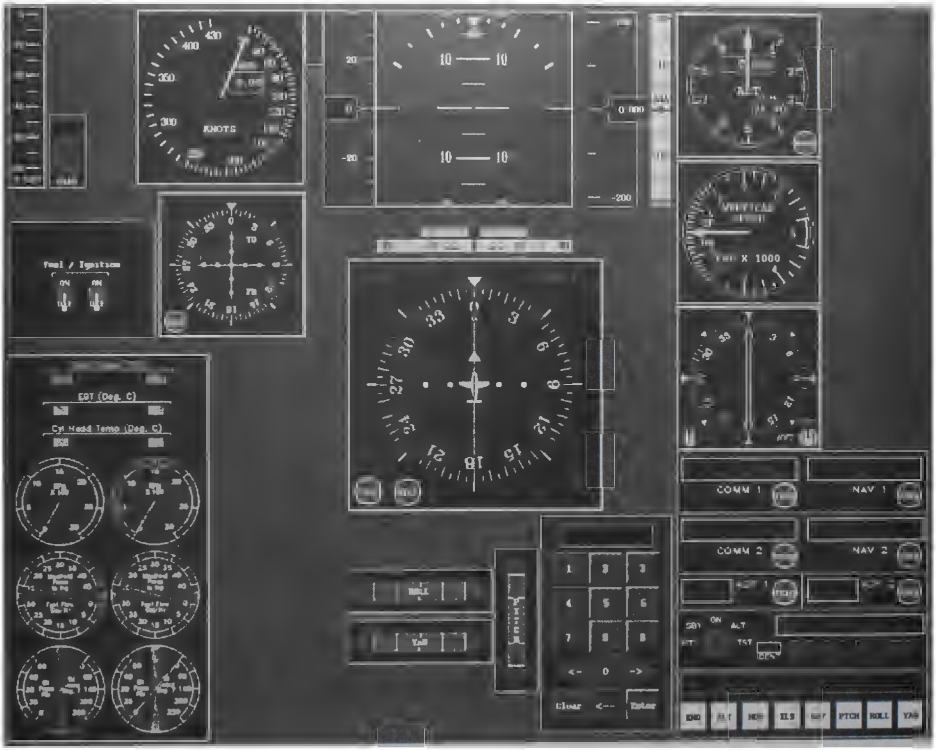


Figure 3. Sample simulated instrument panel. This particular panel is set up for a twin recip aircraft.

AIAA-96-3539-CP

ADS AND THE FOG-OF-SIMULATION: LESSONS LEARNED FROM THE COCKPIT

Herbert H. Bell, Ph.D.
Robert J. Clasen, Capt, USAF

Armstrong Laboratory
Aircrew Training Research Division
Mesa, AZ 85206
(602) 988-6561

Abstract

The Armstrong Laboratory's Aircrew Training Research Division (AL/HRA) has participated in a number of Advanced Distributed Simulation (ADS) exercises. These exercises included Warbreaker, the Synthetic Theater of War-Europe (STOW-E), Multi-Service Distributed Training Testbed (MDT2), and Warfighter 95. Each of these exercises successfully linked large numbers of simulators and enhanced the technical capabilities of ADS. These exercises also suggested that ADS offers unique training opportunities for warriors, battle staffs, and commanders. Unfortunately, these training opportunities were not demonstrated to the same degree as the technology. We believe part of the reason for the failure to fully demonstrate the training potential of ADS stems from problems in adequately defining requirements, appropriately scoping exercises, thoroughly testing simulations, and maintaining real-time exercise control. This paper describes the typical problems we encountered as a node providing virtual simulations of tactical aircraft in support of larger ADS exercises while at the same time trying to provide combat mission training for the pilots flying those simulators. In addition, this paper provides recommendations that we believe will enhance the training value of ADS exercises for participants in simulated weapons systems.

Introduction

Advanced Distributed Simulation (ADS) is viewed as a technology that will enable us to design better systems, develop better tactics, and train better warfighters. Since its beginnings in the Defense Advanced Research Projects Agency's (DARPA's) Simulator Networking project, ADS has shown tremendous technological advances. An Institute of Electrical and Electronics Engineers (IEEE) standard exists for Distributed Interactive Simulation (DIS)[1] and extensive simulation networks capable of creating complex synthetic battlefields have been demonstrated. ADS is clearly the dominant theme in today's simulation and training communities.

We have participated in several ADS exercises including Warbreaker, Synthetic Theater of War-Europe (STOW-E), Multi-Service Distributed Training Testbed (MDT2), and Warfighter 95. One of our roles in each of these exercises was to provide high-fidelity tactical aircraft simulators flown by Air Force pilots. These exercises allowed us to participate in distributed simulation environments with large numbers of other participants and simulations. Each of these exercises was a technological success. Yet each exercise was also an expensive and often frustrating experience for us and our pilots. The primary reasons for this were that the techniques and procedures necessary to effectively and efficiently design, implement, and conduct ADS exercises were, and are still, very immature.

The purpose of this paper is to describe the recurring problems we saw in these ADS exercises as a small virtual simulation node. In addition, we will offer some recommendations that we believe will reduce the effects of some of these problems and enhance the training value of ADS exercises.

Exercise Planning

Requirements Definition

The biggest problem we have encountered in distributed simulation exercises is the absence of well-defined objectives for the human participants. To date, the majority of ADS exercises have focused on creating technologically complex synthetic environments. As a result of this technology emphasis, exercise objectives have typically focused on correcting the technical shortfalls identified in previous exercises and adding new capabilities. Therefore, the objectives tend to emphasize the number of participants, the addition of new simulators, the simulation of new phenomena, and associated improvements in computer and communication technology. Such technological advances are certainly necessary if ADS is to reach its full potential. Unfortunately, in many recent ADS exercises, it often seems as if the human participants in the simulators are merely aids needed to demonstrate the operation of the technology.

We believe the first step in improving the value of ADS exercises is to identify the training audience and the associated training objectives. Exercise planners, training professionals, and technologists must work together to define the training or mission objectives for the individual warfighters and scope the exercise to meet these objectives. Failure to develop such detailed objectives means it is difficult to define the federation of simulators necessary for the exercise. In addition, it is difficult to determine how those simulators should interact with one another and to specify when those interactions should occur. Hopefully, some of these deficiencies will be eliminated as future ADS exercises adopt the Defense Modeling and Simulation Office's (DMSO's) High Level Architecture (HLA). HLA requires formal definitions of all simulation interactions as part of the federation development process.[2]

Once the training objectives and simulator federation have been clearly identified, the exercise planners, trainers, and technical staffs can begin the iterative process of designing an ADS-based exercise. During this phase, the design team must ensure that the simulation scenario is consistent with the training objectives. Once the draft scenario exists, the design team must explicitly review individual simulator capabilities. During this review, weapon system operators (e.g., pilots) must describe the stimuli and responses necessary for them to perform their assigned mission within the draft scenario. The technical experts familiar with the simulations can then describe how these events will appear in the virtual world. Our experience has repeatedly shown that neither DIS protocols nor individual simulators are capable of faithfully reproducing all aspects of the battlefield. In addition, different simulators may reproduce the same aspect of the battlefield at different levels of fidelity. As a result of these discrepancies, changes to individual simulators, DIS protocols, exercise scenarios, or training objectives are typically required. These changes are necessary to eliminate technically impossible events and to identify workarounds to enable other events to appropriately occur. Usually, this review and modification process requires several iterations.

As part of defining the scenario and simulator requirements, the design team must also specify the communications requirements for the exercise. Typically, communication requirements involve both simulation data and voice communications. Simulation data includes the exchange of platform and weapons state information among the participating entities. Voice communication usually includes tactical, exercise control, and technical/simulation troubleshooting nets. In addition, both voice and data communication may be required to support mission planning and after action reviews.

Appropriate Scoping of Exercises

A second problem with distributed simulation exercises, which is related to poor requirements definition, is improperly scoping the size of the exercise. Conventional wisdom seems to be the more entities you can put on the network, the more realistic the exercise. However, "more" is not necessarily "better" when it comes to creating synthetic environments. Large ADS exercises demand large resource commitments during both the development and conduct of the exercise. A large number of entities necessitates an extensive network bandwidth, which can be very costly. Also, the computers used to process the network traffic can become bogged down trying to meet real-time processing demands. In exercises like STOW-E, we have seen network interface units and plan-view displays become so overloaded during peak periods of traffic that they are barely useful. In addition, although the constructive simulations may be able to computationally handle the large number of entities used in exercises such as STOW-E, many virtual simulators cannot realistically process or display such large numbers of entities. Consequently, even though large numbers of entities are broadcast across the network, many virtual simulation nodes can realistically process and display only a tiny subset of those entities. Because these problems can have a negative impact on training, the size of the exercise should be limited to "just enough" entities to make the scenario realistic enough to meet the training objectives

In addition, large ADS exercises require extensive commitments of manpower and facilities to support both exercise development and execution. Most ADS exercises rely on existing legacy systems to provide the weapon system simulators. Since these legacy simulators are already fielded to support other requirements, ADS exercises must compete for personnel and time. This means either extending the period of simulator operation and increasing the number of personnel or reducing support to other programs.

Thorough Testing

Another obstacle to successfully accomplish training in distributed simulation is insufficient testing. Today, most distributed exercises use DIS protocols to allow communication among disparate simulation devices. It is up to each site to implement these protocols correctly; our experience has shown that this is not always done. For example, we recently participated in an ADS exercise in which several sites had major DIS discrepancies. These discrepancies included incorrect versions of the DIS protocols, incorrect entity identifications, incorrect dead reckoning, incorrect timestamping, and errors with PDU fields.

Noncompliance by one site impacts all other sites. At the very least, noncompliance wastes computer resources because each site must process bad data, and, at the worst, noncompliant data causes simulation devices and simulation nodes to "crash." The only way to avoid these problems is to have thorough DIS-compliance testing. During this testing, engineers using the correct DIS tools must analyze the data transmitted from each simulator to determine DIS compliance. Discrepancies must be reported to both the individual sites and to the exercise director. Once the various systems are compliant, the system configuration should be frozen until the completion of the exercise. Unfortunately, this rarely occurs. Typically, due to schedule pressures or last-minute additions of some new capability, testing is incomplete, and a mammoth engineering effort is needed just prior to the start of the exercise to pull everything together.

The first step in the testing process should be for the test director to clearly specify all the DIS Protocol Data Units (PDUs) necessary to exchange information between the sites. The specification must list the PDUs for each site and thoroughly describe any experimental PDUs that will be used during the exercise. These should be clearly documented and distributed to all sites to avoid any possible miscommunication. Armstrong Laboratory (AL/HRA) recently participated in an exercise where a requirement for an Identification Friend or Foe (IFF) PDU existed, but because no one created a list of which PDUs each site needed to support, we were unaware of the requirement. Late in testing, we noticed our F-16 simulators were being shot down by friendly Patriot missiles because they were expecting the F-16s to broadcast certain IFF codes. This problem occurred too late for us to implement the IFF PDUs, and during the exercise the F-16 pilots had to alter flight paths to avoid the Patriots. This "sim-ism" would have been avoided if the DIS PDU support requirements had been explicitly defined.

Once the required PDUs have been defined, each participating site should pass an individual DIS compliance test, like the DIS Test Suite.[3] Once the individual compliance tests are complete, then sites can join the rest of the network on a one-by-one basis to begin interoperability testing with the other sites.

The time spent upfront thoroughly testing individual sites more than pays for itself in the long run. It is much easier to troubleshoot a problem when only one site is involved as opposed to having multiple sites. Also, unexpected PDU data can affect different sites in different ways, making it even more difficult to find the cause of the problem. Individual compliance testing is also a more efficient use of resources. If multiple sites are interconnected and a problem occurs, then personnel at each site must spend time troubleshooting a problem that could have been discovered locally. In addition, simulation resources are wasted when bad data is passed over the network. At a minimum, network interface units must weed out the bad data before passing it along to the local simulation devices. At the worst, non-DIS-compliant data can cause simulations to crash, thus eliminating any training opportunity for the operator of that simulation device.

There are other DIS issues besides being able to support specific PDU types. One of these is update rates. The DIS standard states the default amount of time between transmission of a simulation entity's position is five seconds, unless the dead reckoning tolerance has been exceeded.[4] Often, sites violate this update rate by sending updates at different rates, based on local requirements. Some sites update at a slower rate in order to reduce the amount of traffic on their local area network, but this can cause problems at other sites that are expecting the standard rate. For instance, if the update rates are too slow, other sites will assume the entity no longer exists and will remove it from the list of active entities and all visual displays. Then, when the update does arrive, the site must insert it into the active list again, and the entity will suddenly reappear on the visual displays. This causes entities to pop in and out of the visual displays, resulting in an unrealistic training environment.

Another example of a DIS-related problem that should be checked during testing is the implementation of the Start and Stop PDUs. Many sites implement these PDUs so that all entities on the network are frozen or unfrozen. This is fine on a local area network, but when connected to a wide area network, each site's Start/Stop PDUs should only affect that site's entities. During one exercise, our F-16 cockpits kept unexpectedly coming off freeze and crashing, and we had to keep reinitializing them. This was due to another site which sent out improperly addressed Start PDUs. Problems like this should be caught during testing to avoid wasting valuable training time during the exercise.

Thorough testing is also required to identify and correct fidelity mismatches and incorrect entity identifications. For example, during testing for MDT2, we found that one site was scoring 500- and 2,000-pound bombs as mortar shells. We also discovered that the location of a major road differed by as much as 300 meters between different simulator databases. Without extensive testing, neither of these problems would have been caught before training began.

It is also important to test using the expected amount of network traffic as early as possible. Unfortunately, testing is rarely possible using all the simulation systems in a realistic manner. The primary reason for this is that the warriors who will use these simulators are typically not available until the exercise actually starts.

Such testing, however, is critical for ADS exercises with large numbers of entities. Individual sites may have problems processing the large numbers of entities on the network, and, if the expected network load is not present early enough during the testing period, there may not be enough time to correct the problems. This happened to us during the STOW-E exercise. During testing, we typically only saw 100-200 entities on the network, and we never saw more than 800 entities at one time during the test period. Then, during the exercise, there were 1,700 entities present, and several of our systems choked under the increased processing load. Our plan-view display could not update fast enough, and eventually it crashed, leaving us "blind" as far as seeing the scenario. The network interface units for the F-16 simulators also had a difficult time processing the increased amount of data. One result of these processing problems was that the entities in the pilot's out-the-window display appeared to jump instead of smoothly move within the virtual scene. We may have been able to fix these problems prior to the exercise if we had been able to detect them during testing.

Once DIS compatibility and interoperability requirements are satisfied, the system configuration should be baselined, and no software changes should be made until after the exercise is completed. New problems can arise or old problems that were once fixed can suddenly reappear after a programmer makes one seemingly innocent little change to the software.

Exercise Control

A lack of appropriate control over unfolding events during the execution of the exercise is another reason why distributed simulation has failed to achieve its full training potential. The distributed nature of ADS exercises allows the creation of complex, synthetic environments that offer more realism than previously available; however, having participants scattered geographically also makes it very difficult to orchestrate events in a way to maximize the training benefits.

Of course, it is impossible to have any control over an exercise if there is no means for the exercise director to communicate with the participating sites.

Exercise directors should use a telephone conference call among all participating sites to convey information during execution of the exercise. Currently, the reliability of telephone conference calls is much higher than the reliability of sending digital voice data over the network, and the importance of exercise control necessitates using the most reliable method of communication available. AL/HRA recently participated in an exercise that relied solely on digital voice communication for all voice traffic, and when we lost communication due to compatibility problems with our digital voice unit, we had no way to let the exercise director know that our pilots would not be able to talk with the weapons controller. Flying the mission was a waste of the pilots' and controllers' time, yet we couldn't convey that information to the exercise director.

Even when a reliable communication link is used, sites can still miss important information. It is difficult for each site to have someone constantly monitor the exercise control network--sometimes personnel must walk away to reset a computer or possibly answer another phone. During these gaps, if the exercise director announces, for instance, that the network will be down for an hour, not all sites may get the information. These sites may have pilots in the cockpit waiting for an expected mission to begin. To avoid this type of situation where trainees are potentially wasting their time, exercise directors should use a roll call when announcing information that is of importance to all sites. By going through the list of participants one by one, the exercise director can ensure that everyone is listening and gets the word.

The exercise director should also use the control network to closely monitor execution of the scenario. During the confusion of large-scale exercises, it is easy to forget about individual sites, but in order to maximize the training potential, the exercise director must know whether the scenario is unfolding according to plan. For example, AL/HRA participated in an exercise where our F-16 pilots were to fly as part of a strike package. Enroute to the target, we realized we had no escort and that none was forthcoming. We later learned that the originating site of our escorts had network problems and was not able to carry out its mission. If the exercise director had been aware of this and had let us know, perhaps we could have delayed the mission or at least exercised command and control procedures to inform the flight of the status of the escorts. This would have offered the opportunity for some useful training. As it was, everyone associated with this mission missed a valuable training opportunity and the F-16 pilots flew a meaningless mission.

Recommendations

In order to maximize the training benefit of advanced distributed simulation exercises, we recommend program managers follow these guidelines:

1. First and most important, develop a list of comprehensive training objectives.

2. From the training objectives, derive the scenario, communication plan (data and voice), and list of potential participating sites.

3. Define exercise management requirements, to include security, exercise control, mission initialization, and after action reviews.

4. Review site capabilities and ensure that assigned missions are within the capabilities of that site's simulators.

5. Finalize scenario, simulation data requirements, communication requirements, and the exercise schedule.

6. Develop a comprehensive testing plan.

7. Conduct rigorous testing, and then establish a baseline for the system.

8. Pretrain and rehearse participants.

9. Conduct the exercise.

10. Archive exercise baseline, exercise data, and document lessons learned.

Conclusion

Based on our observations, we believe that ADS has the potential to significantly enhance combat mission training. However, in order to fully realize this potential, exercise planners and trainers must work together to define the training requirements. Once the training audience and training objectives have been defined, the appropriate simulation technologies and training strategies should be identified. Failure to adequately define training requirements, identify simulation resources and training strategies, and test system components leads to the fog-of-simulation in which resources are ineffectively used and trainees become the fodder for technology demonstrations.

References

1. Institute of Electrical and Electronics Engineers, *IEEE Standard 1278.1, Standard for Distributed Interactive Simulation -- Application Protocols*, 1995.

2. Dahmann, Ponikvar, and Lutz, "HLA Federation Development and Execution Process," *14th Workshop on Standards for the Interoperability of Distributed Simulations*, March 11-15, 1996, pp. 327-335.

3. McAuliffe, Long, Liu, Hays, and Nocera, "DIS Test Suite," *14th Workshop on Standards for the Interoperability of Distributed Simulations*, March 1996, pp. 561-566.

4. Institute for Simulation and Training, *Standard for Distributed Interactive Simulation--Application Protocols, Version 2.0, Fourth Draft (Revised)*, March 1994, p. 25.

EVALUATION OF MODSAF AIR ENTITY SIMULATION

Michael Conquest and David Lerman
Hughes Training, Inc., Training Operations
Mesa, Arizona

Herbert H. Bell
Armstrong Laboratory,
Aircrew Training Research Division
Mesa, Arizona

Abstract

Advanced Distributed Simulation (ADS) offers the potential for providing composite force training for aircrews using ground-based simulators. Such composite force training, however, will most likely require a robust set of computer-generated forces (CGFs) to complement manned simulators. CGFs are necessary to provide the large number of aircraft needed to realistically depict composite force missions. There are a number of different CGFs available at the present time. Each of these CGFs was developed to meet a specific need and has its own unique strengths and weaknesses. This paper describes the Modular Semi-Automated Force (ModSAF) system's capability to provide realistic computer-generated aircraft for use in ADS exercises. This evaluation looked at selected flight performance and mission behaviors as well as aerodynamic and flight dynamic modeling. Results indicate that ModSAF provides only a rudimentary capability to realistically depict modern fighter aircraft. Deficiencies were identified in mission behavior, aerodynamic modeling, and weapons loads. Although the ModSAF software architecture is sound and well organized, incorrect modeling of aircraft and mission behaviors presents serious limitations in using ModSAF to represent current Air Force weapons systems. Recommendations include a complete redesign of the fixed-wing aircraft algorithms and data files as well as an expanded set of mission behaviors.

1.0 Evaluation Objectives

This evaluation was conducted at the request of the Electronic Systems Center, Hanscom AFB, MA. Its purpose was to assess Modular Semi-Automated Force (ModSAF) capabilities as a computer-generated force (CGF) system for realistically depicting airborne weapon systems in Advanced Distributed Simulation exercises. This evaluation used both simulated mission tasks and system analyses to determine the weapon system's flight performance characteristics and doctrinal behaviors, as well as the overall ModSAF system performance. No attempt was made to compare ModSAF capabilities to other CGF systems.

2.0 Air Entities

ModSAF 2.0 has a very limited number of aircraft types modeled. This evaluation focused on the F-16D because it is the only USAF fighter aircraft modeled in ModSAF 2.0. In addition, because the Armstrong Laboratory, Aircrew Training Research Division, conducts training research and development using F-16 aircrew training devices, F-16 pilots, simulation engineers, and two Simulator Certified F-16C Weapon System Trainers (WST) were available to support this evaluation.

Although the F-16D was selected, the ModSAF depiction was inadequate to complete all evaluation areas. First, the "D" model of the F-16 is not a particularly good selection to represent the F-16. The F-16D is a two-seater primarily used for F-16C training. With the exception of F-16 training wings, there are only a few F-16Ds at each base operating F-16Cs. The F-16C would be a much better model to represent the F-16 aircraft in ModSAF. Next, the F-16 is a multi-role fighter, capable of air-to-air and air-to-surface missions. ModSAF limits the F-16D to the air-to-surface role. The weapon load selections available for the ModSAF F-16D are inadequate and in some cases incorrect. ModSAF correctly allows you to load AGM-65 (Maverick), AIM-9 (Sidewinder), and Mk-82 bombs. ModSAF also allows you to load 2,000 M-50 series 20mm gun rounds. This is incorrect since the F-16 has a 20mm M61A1 cannon capable of carrying 512 rounds. Also, ModSAF should include other weapons, such as AIM-120 (AMRAAM); area munitions such as CBU-87; Mk-84 bombs; as well as laser guided bombs such as GBU-10 and GBU-12. In addition, ModSAF does not distinguish between the different variants of the same munition, e.g., AGM-65A versus AGM-65D, nor different variants of the same aircraft, e.g., F-16C/D Block 30 versus Block 40.

Because the ModSAF F-16D is limited to air-to-ground behaviors, the ModSAF F-14D was used to evaluate air-to-air task behaviors. In addition, the MiG-29 was used for tests requiring an opposing force (OPFOR).

3.0 Tests

All tests were conducted using ModSAF 2.0 as distributed without modification. ModSAF Kits A and

C distribution documentation as well as on-line documentation were used to build and run the test scenarios. The software was run on two Silicon Graphics Inc., (SGI) workstations. One Indigo 2 Extreme ran as the SAF simulator while an Indy served as the SAF station. All models were completely dependent on ModSAF 2.0 software. All aircraft behaviors were controlled through the use of ModSAF Task Frames with no additional intelligent forces interface such as SOAR (taking a State, applying an Operator, And generating a Result). Additional test tools included the F-16C WSTs and a Sun 4 workstation for network data collection and analysis. Distributed Interactive Simulation (DIS) protocol 2.04 was used for all network-based testing.

Tests were designed to determine the fidelity of the ModSAF F-16D's aerodynamic and behavioral representation. Climb, turn, and acceleration tests were developed to determine aerodynamic fidelity. Scenarios were also built to test the threat detection, bingo fuel, Winchester weapons, and terrain following capabilities of ModSAF. The following sections describe these tests and the results.

3.1 Climb Performance

A number of tests were conducted to determine the F-16D's climb performance with different weapon system configurations. After initial results showed that the desired outcome could not be achieved, other tests were added replacing the F-16 thrust map with the default F-16D parameters which surprisingly include the F-14D thrust map.

Figure 3-1 shows the climb performance test scenario. The test began with the F-16D flying an ingress route at .87 Mach, 3,000 ft MSL. Upon completion of the initial leg of the ingress, the F-16D was tasked to perform an immediate climb to 33,000 ft. For these tests, all weapons were removed and the fuel set to 6,294 lbs.

F-16 Thrust Map Climb Performance

This test scenario was initially run using the F-16 thrust map in the fixed-wing configuration menu. All other configurations remained at the default settings. The F-16D could reach only 4,541 ft in altitude. A closer look at the data revealed that the aircraft started out at 290 kts, the default take-off speed, and could not accelerate up to the commanded ingress speed of .87 Mach (560 kts). When the aircraft attempted to climb for the second phase of the ingress, it eventually stalled and crashed.

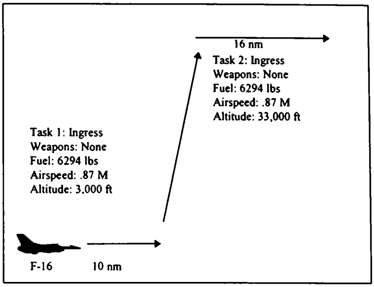


Figure 3-1. Climb Performance Task

A second attempt was made, increasing the fixed-wing aircraft configuration take-off speed to 560 kts, same as the commanded ingress speed. This time the F-16D was able to achieve 19,554 ft altitude, but again stalled after losing all airspeed.

The test scenario was modified a number of times, moving the ingress tasks apart to command a slower climb and using various take-off speeds, all with similar results. Each time, the aircraft attempted to maintain a steep climb angle until it stalled. Then it rapidly went into a steep dive and could not recover.

F-14 Thrust Map Climb Performance

Since the initial test using the F-16 thrust map showed the ModSAF F-16D unable to complete the test, we repeated the test using the F-14 thrust map. The F-14 thrust map is the default for the F-16D entity. No documentation could be found telling the user to change the thrust map configuration when creating an F-16D.

The F-16D climbed beyond the commanded 33,000 ft to over 45,000 ft. It achieved 33,000 ft in approximately 30 seconds. With the F-14 thrust map, the ModSAF F-16D climbed much faster and maintained airspeed longer in the climb than an F-16 aircraft.

F-16C Weapon System Trainer Climb Performance

To verify that the ModSAF F-16D using either of the thrust maps produces unrealistic flight performance, Armstrong Laboratory's F-16C Weapon System

Trainer was configured the same as the ModSAF F-16D, and the climb performance was recorded. The F-16C WST was able to achieve 33,000 ft in approximately 60 seconds. The F-16C WST reached 33,000 ft within the distance between the end of the first ingress phase and the beginning of the second ingress. However, the F-16C could not maintain the steep flight path angle attempted by the ModSAF F-16 test. The F-16C WST had to gradually decrease the flight path angle to maintain airspeed during the climb. As previously noted, the ModSAF F-16s maintained the steep flight path angle until they stalled. Figure 3-2 illustrates the climb performance of the ModSAF F-16D with both thrust maps and the F-16C WST.

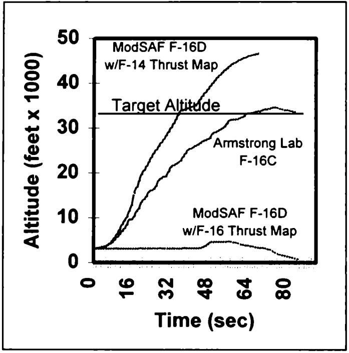


Figure 3-2. F-16 Climb Performance Comparison

3.2 Effects of Weapons/Fuel Load on Performance

Additional Climb performance tests were conducted to determine the effects of weapons load and fuel on the ModSAF F-16D flight performance. The previous climb test scenarios were used but with the take-off speed modified to 560 kts to achieve the same initial speed during all tests when entering the climb. The F-16 thrust map was used on all three tests.

The first run was conducted with the F-16D configured with no weapons and 6,294 lbs of fuel. It was able to achieve 19,554 ft of altitude before stalling. The second configuration increased the fuel load to 12,884 lbs with no weapons. In this configuration, the F-16D could reach only 17,035 ft before stalling. Finally, we added weapons: 4 Mavericks, 2 Sidewinders, 6 Mk-82s, and 2,000 M50s with the 12,884 lb fuel load. The F-16 reached 17,039 ft in this configuration.

Figure 3-3 shows that the ModSAF F-16D's fuel load does impact climb performance. It also shows that the ModSAF did not model the additional drag and weight that weapons add to the aircraft.

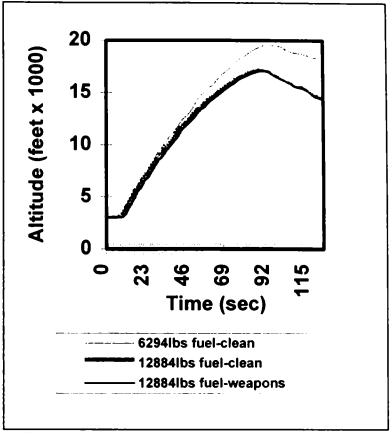


Figure 3-3. Climb Performance Variation Due to Configuration

3.3 Turn/Acceleration Testing

Test scenarios were designed to determine turn and acceleration performance. However, the incorrect thrust modeling caused the turn rate tests to be inconclusive. The turn data collected did not provide turn information that could determine whether or not the turn performance was correctly modeled. While using the F-16 thrust map, target speeds could not be obtained for comparison to the flight manual data. It was also difficult to determine if the ModSAF aircraft was attempting a max rate turn, a standard rate turn, or something in between. Acceleration tests were not conducted because the data collected during climb performance clearly indicated incorrect thrust or drag modeling.

3.4 Threat Detection - Air

A ModSAF sweep scenario with the MiG-29 was created, networked to the manned F-16 WST for testing the air-to-air threat detection capabilities of ModSAF. The MiG-29 was placed on a straight sweep route with the radar on +/- 40 degree azimuth. The

ModSAF status window was used to determine when the MiG-29 was aware of the F-16.

The F-16 WST was initialized approximately 10 nm in trail of the MiG-29. The F-16 locked on the MiG-29 with radar and continued to close without detection by the MiG-29. This result was consistent with the ModSAF documentation of the detection model but is not realistic. This reflects the absence of a standardized emission Protocol Data Unit (PDU) in the Distributed Interactive Simulation communication protocol. Such a standardized emission PDU is required before ModSAF can effectively incorporate a radar warning receiver detection model.

The F-16 WST then moved wing-to-wing with the MiG-29. Even though the F-16 was well within visual range, the MiG-29 did not detect the F-16. The status page would intermittently change to "avoiding collision with nearby aircraft," but never detected the F-16 as a threat. This verified documentation which indicates the ModSAF does not consider visual threat detection during air-to-air engagements.

Once the F-16 overtook the MiG-29 and was within its radar volume, the MiG-29 detected the threat. The F-16 continued to approximately 10 nm ahead of the MiG-29 and then turned back toward the MiG-29. The MiG-29 then began targeting and firing on the F-16 consistent with the user-entered rules of engagement and commit criteria.

3.5 Threat Detection - Ground

Figure 3-4 shows the test scenario used to determine how the ModSAF F-16D would react to ground threats along a pre-planned route. Two flights of F-16Ds were given a low-level ingress route to a target area. An SA-9 platoon was placed along the route. The F-16D flights were separated by approximately 30 seconds to determine if any communication or command and control is modeled that would allow the second flight to alter their behavior based on information obtained from the lead flight.

The scenario was first run with the SA-9 weapons on hold. The lead flight aircraft detected the SA-9 site at about 4 miles out (consistent with the input visual range parameters) and continued on route with no evasive maneuvers. The trail flight followed the exact same route.

The scenario was repeated with the SA-9 site on weapons free. Again the F-16Ds detected the SA-9, but continued on route and were shot down by the SA-9s. The trail flight continued and was also shot by the SA-9s with no evasive maneuvering.

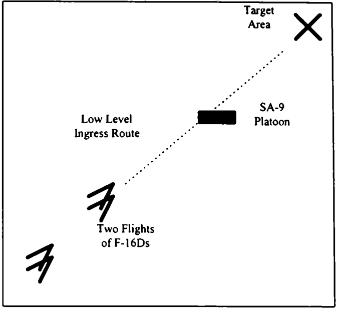


Figure 3-4. Ground Threat Detection Scenario

3.6 Bingo Fuel

An F-14D entity was used to test ModSAF's bingo fuel task. The F-14D was assigned a sweep task with 2,500 lbs of fuel. The status page indicated that 2,366 lbs of fuel were required to return to base (RTB) for the selected bingo fuel point. The F-14D flew over 80 nm, and the status page never indicated a decrease in fuel. Switching over to the entity creation page showed that the fuel had decreased to 2,273 lbs, 93 lbs less than needed to RTB. The F-14D continued on the sweep mission. The F-14D would transition to RTB only after the operator manually entered a lower value of onboard fuel on the entity creation page while the aircraft was actually flying the sweep.

3.7 Winchester

The F-14D was also used to test whether or not the aircraft would correctly RTB if it ran out of weapons during a sweep mission. The F-14D was initialized with only two sidewinders. Five MiG-29s were flown head-on to the F-14D. The F-14D fired both its missiles at the MiG-29s. Although the F-14D had no remaining weapons and was aware of the MiG-29 threats on its nose, it continued its sweep making no attempt to RTB. The test was repeated with the F-14D initialized with no weapons and produced the same results.

3.8 Terrain Following

Terrain following was tested by assigning a ModSAF F-16D ingress task over a section of the National

Training Center database. The terrain varied from 1,800 ft to over 6,000 ft along the flight path. The model varied altitude along the route and avoided the terrain. Climb rates, terrain clearance, and G loading along the route were not analyzed.

4.0 Evaluation Of ModSAF Flight Software

Some of the fwa_ and hm_ software files were reviewed along with two pieces of documentation from the Programmer's Reference Manual and the on-line monitor displays of parameter default initial values. The documentation reviewed consisted of seven pages of parameter initialization (US_F-16D parameters) and a condensed single-page overview of LibFwa that contains many obvious typos causing equation errors.

An examination of the documentation and code suggests that it is based on a limited understanding of aerodynamics and mechanics. The code is almost impossible to read or maintain because many variables are given names or definitions that are inconsistent with their application in the algorithms, the comments are not illuminating, and the units are rarely defined. In addition, the effects of many of the algorithms are inconsistent with the physics of the real world. The sparse top level documentation does not always match the code and is insufficient to describe the architecture or functions. A few examples of these problems are presented in the following sections.

4.1 Axis Systems, Angles, And hull_to_world Matrix

Function fwa_tick in file fwa_tick.c calculates a matrix called hull_to_world. This is calculated in terms of the trigonometric functions of the roll, track, and flight path (climb) angles. Angle of attack (AOA) is used in the calculation of many flight characteristics, as described later, yet no account is taken of its effect on the attitude of the hull relative to the flight path and thus relative to earth. Does the omission of AOA from this matrix mean that although the effect is accounted for in aircraft performance, it is not to be represented in the aircraft attitude?

Analysis of much of the software suggests that roll, flight path angle, and track have their conventional definitions with respect to map Grid North, Grid East, and Down. Analysis based upon this shows that the matrix is world_to_hull, not hull_to_world. Furthermore, the world axes are Grid North, Grid East, and Up and the hull axes are Left, Forward, and Top. Both these sets of axes represent unconventional and left-handed systems. Relative to these axes, roll, flight path angle, and track are not Euler angles. This matrix is used in many places, but the use was not evaluated.

Immediately after the matrix calculation, the X, Y, and Z components of earth axis velocity are calculated from the speed and the trigonometric functions of flight path angle and track. The equations used confirm that the world axes correspond to Grid North, Grid East, and Up. Further confirmation of the angle and axis conventions is provided elsewhere by track corresponding to atan2 (Y_vel, X_vel).

Even with knowledge of the unconventional axis systems, to use a standard matrix vector multiply function to obtain the world axis velocities would be wrong, since the matrix is named as hull_to_world, the inverse of its real meaning.

The transformation between map grid and geodetic coordinates has not been analyzed with respect to the suitability of the map projection or proper accounting for meridian convergence when transforming heading, velocity, and acceleration between systems.

4.2 Equations Of Motion

Track, flight path angle (climb), and roll angle are integrated from their individual rates of change, a poor method, especially near the vertical. The roll, track, and flight path angles are each expressed in the range $-\pi$ to $+\pi$ through the application of the function angle_clip. This constraint is proper for roll and track, but totally wrong for pitch angle which should never exceed $\pi/2$. The integration of pitch angle near the vertical needs special care, and it is hard to visualize the functioning of the simulation at a declared pitch angle of 120^0.

Speed by definition is along the flight path and is integrated from speed rate.

4.3 Sideforce And Sideslip

In the code, the contributions of sideforce to fpa->rate and track->rate have the wrong sign, unless sideforce is defined as positive to the left, a frightening and confusing thought, although consistent with what we learned about the axes through analysis of the hull_to_world matrix. The documentation of LibFwa shows sideforce in the flight_path_angle_rate calculation, but it is omitted from the track_rate.

Although sideforce affects the attitude rate, the contribution is ignored in much of the code, as when it calculates desired forces, maximum turn performance, limits, etc. The calculation of side force is not shown in the documentation, nor was it found in the code. Sideslip, which might be expected in the presence of sideforce, is not modeled. Those rare occasions when a pilot might apply sideforce and sideslip produce a complicated aerodynamic and artificial intelligence situation which requires an extremely high level of sophistication for an unmanned simulation. The rarity of the application of sideforce and the difficulty of appropriately modeling it suggest that there is little reason to include sideforce in CGF.

4.4 Second Order Control Of Angle Of Attack

An angle of attack goal is calculated from the desired lift. Angle of attack, and thus lift, is controlled as a second order system with a damping ratio of 0.8. The comments mention damping ratio Eta, whereas the conventional name is Zeta. More seriously, very complicated difference equations are implemented. Since the selected properties of the system were arbitrary, it seems more reasonable to use simple and understandable difference equations that give a very close approximation to the correct solution provided the selected natural frequency is slow compared with the computation interval.

4.5 Second Order Roll Control And Effect Of Limits

Roll angle is controlled toward the roll goal with second order equations similar to those used for angle of attack, but this time the system has critical damping. Roll->actual is integrated from the roll->desired_rate, which itself is set after the angle integration.

Roll rate is not explicitly limited in this function. However, function fwa_limits contains the line:
roll->rate = roll->desired_rate, limited between +/-π.

Since the value of roll->rate does not affect roll->desired_rate, the function has no effect on the second order control of roll angle.

An additional problem with the roll rate limits is that they are set constant at -π to +π. Although these are reasonable values for unsophisticated limiting of body axis roll rate, the roll angle being integrated is the roll relative to earth. This may have a large rate of change near the vertical, even with a low body axis rate. Appropriate limits for the rate of change of roll angle need to be calculated and applied in real time, or the whole attitude integration system should be changed.

4.6 Max Turn Rate And Effect Of Limits On Rates Of Change Of Track And Flight Path Angle

Function get_turn_performance sets
normal accel = max aero lift / mass.

Then it sets,
max turn = normal accel/speed.

No structural loading limits are applied, nor is account taken of the resolution of thrust through angle of attack or of the possible application of sideforce although it is modeled elsewhere. In addition, the fundamental relationship to normal force is wrong. As occurs with much of the limit code, there is confusion between loading normal to the flight path and acceleration; the effect of gravity on the flight path is ignored. What the code calls normal acceleration is, in fact, normal loading in units of acceleration. The turn performance relation should be:

max_level_turn = sqrt(normal_loading^2 - G^2)/speed; (if real)

In function fwa_control, the desired track rate is derived from the track error, then bounded in terms of the current maximum lift with the implied assumption that all the loading or lift is available for turn acceleration in the horizontal plane. However, in practice some of the lift is required for acceleration or equilibrium in the vertical direction. Similarly, the desired flight path rate is bounded without reference to turn requirements or the need to oppose gravity. No account is taken of the possible application of sideforce.

Track_force and fpa_force are calculated from the improperly limited desired track and flight path angle rates. The roll->goal is calculated from these forces after further adjustment and limiting through the total normal force. The roll goal ensures that the (limited) desired flight path rate is achieved, and what lift remains is used to generate a modified desired track rate. Since the calculation started with incorrect assumptions, it produces an incorrect answer.

Much of the trouble with the application of the limits results from starting with separate desired track and flight path angular rates. If it started with a single desired normal acceleration, the application of limits would be simpler.

In the above procedure, a further error was found with the calculation of the flight path angle force. This force is normal to the velocity and in the vertical plane containing the velocity. It is calculated as:

fpa_force = mass * speed * cos(roll->actual) * fpa_desired rate + mass * G * cos(fpa->actual)

The cos (roll->actual) should not be in this equation.

4.7 Properties Of The Atmosphere

File hm_utils.c contains four functions representing properties of the ICAO standard day:

- hm_air_density (altitude)
- hm_air_density_rat_sq (altitude)
- hm_mach_velocity (mach, altitude)
- hm_velocity_mach (velocity, altitude)

The last two functions are correctly and adequately described in the comments, and make a good approximation to converting speed, in m/sec, to mach or the reverse, as a function of altitude in meters. This is very reasonable, since ModSAF is mainly a metric simulation.

The first two functions generate reasonable approximations to ambient density ratio and its square, respectively. The comments do not describe the functions or their units. The first function name suggests that it calculates actual air density (ρ), rather than what it does calculate, density ratio (σ), which is the ratio of ρ to the standard value at sea level, ρ_0. However, the name of the second function correctly leads to the expectation that it calculates σ^2.

Numerical analysis of both of these density related functions shows that for the output to be correct the input altitude must be in feet, despite this being a metric simulation.

Function fwa_tick(...) in file fwa_tick.c contains the following line:

fwa->air_density = hm_air_density(position[Z]).

The above is both misleading and wrong. Position [Z] is in meters, yet the function needs feet as an input, as described earlier. Thus the function output is seriously wrong. Also, the function calculates density ratio, but the value is assigned to something called air density. This error of calculating density ratio (σ) but assigning it to something called air density or rho occurs throughout the simulation.

The one page overview of LibFwa states:

air_density = initial_air_density * (the expression for density ratio, σ),

where

initial_air_density = .0249

Thus, unlike the code, the documentation does include ρ_0, but in what units? The correct value is .00237688 slug/ft^3 or 1.225 kg/m^3 in English or metric units, respectively.

4.8 Lift, Lift Coefficient, And Lift Curve Slope

Function fwa_control generates an aerodynamic lift limit which is based on an angle of attack limit α_{limit}. This angle of attack limit must be in radians to be consistent with other lift and drag software. With some nomenclature simplification, the lift limit is set by:

curr_lift_max = .5 * speed^2 * fwa->air_density * fwa->params->lift_coef * α_{limit}

Rearrangements of the above relationship occur throughout the software.

It has already been shown that fwa->air_density is, in fact, the density ratio (σ). Therefore, for the above expression to be dimensionally and physically correct, the variable fwa->params->lift_coef must represent the product of wing area (S), air density at sea level (ρ_0), and lift curve slope (C_{L_α}):

fwa->params->lift_coeff = S * C_{L_α} * ρ_0.

For an aircraft like the F-16, the lift curve slope should be around 3.5; for no aircraft should it be greater than 2π. Knowing that the F-16 wing area is 27.8709 m^2 (300 ft^2) and that the sea level air density is 1.225 kg/m^{3,} we can calculate the simulated lift curve slope from the above expression when we know the value of fwa->params->lift_coef. An on-line initialization page shows that a coefficient of lift is set at 404.13, and this value is also shown in the parameter value documentation. This yields a C_{L_α} value of 11.8, which is several times larger than is realistic.

The documentation overview of LibFwa gives a different relationship that may be summarized as:

Coef_lift = 21522.3
initial_air_density = .0249

$$\text{lift} = .5 * \sigma * \text{initial\_air\_density} * \text{speed}^2 * \text{coef\_lift} * \text{AOA}$$

Substituting the real-world relation:

$$\text{lift} = .5 * \sigma * \rho_0 * \text{speed}^2 * S * C_{L\alpha} * \text{AOA}$$

into the above gives:

$$\text{coef\_lift} = S * C_{L\alpha} * \rho_0 / \text{initial\_air\_density}$$

Substituting metric numerical values into the above and rearranging gives: $C_{L\alpha} = .0249 * 21522.3 / (1.225 * 27.8709) = 15.696$ per radian. This value is 33% larger than the value derived by analyzing the code. Both values are several times larger than reasonable.

Although the meaning of coef_lift differs between the code and the LibFwa documentation, in neither case does it correspond to what aerospace engineers mean by lift coefficient, C_L, which is defined as lift/$0.5\rho V^2S$.

4.9 Drag Due To Lift

The drag due to lift model in function fwa_drag is based on the assumption that the resultant force due to lift acts close to the aircraft Z direction. Because the oversized lift curve slope yields too small an AOA, this model gives an induced drag value that is many times too small.

Even if the AOA were properly calculated, this would be a very poor induced drag model. A much better approach is to calculate C_{D_i} as a function of C_L specific for each aircraft type.

4.10 Drag At Zero Lift

The zero lift drag model in function fwa_drag corresponds to:

```
coef_drag = fwa->params->missile_coeff +
   fwa->params->airplane_drag_index.
```

Subsonic drag is:

$$\text{drag} = \text{fwa->air\_density} * V^2 * \text{coef\_drag} / 2;$$

An analysis similar to that for lift, remembering the air density term actually is density ratio, σ, shows that:

$$\text{coef\_drag} = S * C_D * \rho_0$$

The parameter documentation states that the drag index is set to .83313, which is therefore the value of coef_drag without missiles. From the above relationship this corresponds to:

$$C_D = .83313 / (1.225 * 27.8709) = .0244.$$

Such a value is believable at low Mach numbers, but it is of unknown accuracy.

Supersonic zero lift drag is calculated as 1.06 * the subsonic value. This is poor. The model should be a continuous function of Mach number. Also, the transonic drag rise is far too small.

The meaning of coef_drag does not correspond to what aerospace engineers mean by drag coefficient, C_D, which is defined as drag/($0.5\rho V^2S$).

4.11 Thrust

Function hm_thrust calculates maximum thrust for a given flight condition, reading the value from a table of thrust versus speed and altitude.

If speed and altitude are off the table, the function multiplies the first thrust value in the table by σ^2. The square is a poor thing to do. If this square was the only reason for having a function that generates σ^2, the function, hm_air_density_rat_sq is not needed. The comments claim that this calculation gives the minimum thrust necessary to counteract drag, yet drag was not considered nor does it appear relevant.

The comments claim that little would be gained by interpolating on the table, so the nearest value of thrust in the table is used. Even though the data may be poor, it would be preferable to interpolate between data points to ensure max thrust changes smoothly with flight conditions.

It would be preferable to tabulate thrust/σ rather than thrust, because the value would change much more slowly with altitude, permitting fewer data points. It would also be preferable to follow standard conventions and use Mach Number rather than speed for the second independent variable.

4.12 Fuel Mass Units And Density

Function fwa_get_current_mass has the following single executable line of code:

```
return(fwa->params->vehicle_mass +
       fwa->fuel * fwa->ppl);
```

Is the ppl pounds per liter? Is fuel in liters? Is mass in kg? An initialization page shows fuel density set at 6.8 pounds per gallon, but what is the value of fwa->ppl?

4.13 Observations Based Upon Documentation Of US_F16D_params.rdr

Extracts from the documentation of F-16D parameters, US_F16D_params.rdr, are given below with additional comments.

(airplane_drag_index .83313) ;; 132.4
(airp_drag_index) * 0.0249 air_dens_msl

Once more, .0249 is not the correct value for ρ_0. The arithmetic is not correct; the product should equal 3.29676, nearly four times greater than the value assigned. What actually got into the code? It is a pity the physical meaning of the term is not explained.

(vehicle_mass 9100.00) ;; kg . This is 20062 lb, excluding fuel and stores.

(lift_min -177,956.0) ;; N, corresponds to -2G structural limit.

(lift_max 800,803.0) ;; N, corresponds to +9G structural limit.

The above structural lift limits correspond to the stated G limits at the very low empty weight. Thus, if applied, they seriously curtail maneuverability at higher weights regardless of the available aerodynamic lift. It would be far better to specify structural limits in G, possibly as a function of stores, and calculate the corresponding lift limits in real time as a function of weight.

There was no reference to limits on Mach or equivalent airspeed.

4.14 Absence Of Function To Perform Linear Function Interpolation (LFI)

Function fwa_fuel_usage_rate calculates fuel flow as a function of altitude and percent engine performance by linear interpolation on a two-dimensional table of data. The process is coded specifically for this fuel flow function, with frequent references by name to the input variables and properties of the referenced data table. The process does not use a general purpose function for performing Linear Function Interpolation (LFI).

As described earlier, function hm_thrust calculates maximum thrust from a two-dimensional data table without using linear interpolation; it uses the nearest value in the table. Like the fuel flow, this process is coded with specific reference to the names of the input variables and data table.

Taken together, these two pieces of information strongly suggest that there is no general purpose LFI function. If there had been a general purpose function, it would have been used on the fuel flow and the thrust; there would be no need for comments explaining why interpolation was not performed for thrust. It also explains why many other properties are over-simplified, rather than being modeled as LFIs.

4.15 General Observations On Flight Model

The preceding sections have addressed just a few of the problems in the flight and control model software algorithms. Most of the other software examined has shown a low level of expertise in the algorithms similar to that highlighted herein. The lack of the usual axis or sign conventions, poor math and physics, misuse of nomenclature, carelessness with units, and inadequate documentation make the aerodynamic software hard to understand, correct, maintain, or use.

In some respects the code has been over modularized, with similar relationships occurring in many modules. For instance, the lift relationship or its inverse is repeated or implied in many functions, all of which must be modified if the lift relationship is changed. Some functions use or imply relationships slightly different from others, which may cause errors. For instance, a stall speed equation ignores the contribution of thrust resolved through the angle of attack. Another example is that some functions use sines and cosines when resolving thrust through the AOA, whereas others use small angle assumptions. As already mentioned, the effect of sideforce is included in some functions but not in others.

5.0 ModSAF Architecture

This section evaluates ModSAF architecture based on the software design methodology, without regard for the specific implementation of aerodynamic and flight algorithms described above. The "Software Architecture and Overview Document for ModSAF"* distributed with ModSAF details the design methodology. The document describes four techniques used in the development of ModSAF.

* Software Architecture and Overview Document for ModSAF, Version 2.0, Sep 29, 1995

- Layering
- Object-Based Programming
- Rigorous Interface Specification
- Data Driven Execution

These techniques provide the basis for a sound software development strategy. Adherence to these techniques eases the management of parallel software development and test activities. The resulting modular software facilitates growth and modification of libraries for future ModSAF releases, as well as user enhancements for specific site requirements. Data driven execution provides the means to add new vehicle types with minimal software changes. New entities may be added through the modification of parametric data files.

To take advantage of data driven execution, the underlying ModSAF algorithms and parametric data must be capable of fully supporting the entities' performance and behavioral characteristics. As noted in Section 4.0, the algorithms and parametric data do not accurately depict fixed-wing aircraft and they make it difficult to add new aircraft types.

The latest distribution of ModSAF, version 2.0 contains 394 software libraries with over 600,000 lines of C source code. The software is intended to be portable to most Unix workstations supporting X windows and Motif for the Graphical User Interface (GUI) operation. The software is also intended to be operating-system independent by selecting the correct platform and operating system options for compilation. Although the system has been run on many different systems, portability seems to be an issue based on ModSAF reflector messages.

ModSAF is a real-time, entity level simulation. Each simulated entity's state is updated periodically using a variable period (or tick). During low levels of activity, ModSAF updates each entity every 67 msec. As the activity increases, the ModSAF update period lengthens to accommodate all locally simulated and remotely generated entities within each processing period. This allows the system to handle spikes in processing requirements. The software allows the user to fully utilize the available processing capability on the ModSAF host based on a nominal load. If spikes in processor requirements occur, the system gracefully handles the increased requirement by slowing down until the requirements again normalize.

The number of entities simulated, network traffic, types of entities, activity of the entities, and surrounding terrain are some of the factors that affect update rate. Also, the required update rate for entities varies based on the use of the simulation.

ModSAF benchmark tests are run using the criterion that each entity over the course of one minute, must 90% of the time, be updated every 500 msec. Using this criterion, the developers determine the maximum number of active ground entities a specific version of ModSAF can simulate. This gives them a basis for determining how software changes impact processing requirements at the system level. It also provides a method for determining run-time efficiency across hardware platforms.

Although this criterion is adequate for engineering comparison and may be suitable for ground force interaction, much faster update rates are required to generate air entities flying with or engaging a human-in-the-loop virtual simulation. As the system slows down, entities start jumping, making it difficult for a manned simulator to engage air entities within visual range. As the system slows down further, the manned simulation begins to lose radar lock during beyond-visual-range engagements.

A test was run using an SGI Indigo2 with a MIPS R4400/200 MHz processor with 128 MB of memory as the SAFsim. An SGI Indy was used with a MIPS 4600/134 MHz processor and 96 MB of memory as the SAF station. Both machines were running IRIX 5.3 operating system. With no network traffic, no engagements, ModSAF F-16Ds flying ingress routes began slowing the system down at about 20 entities. At 60 entities the system update rate had doubled.

When running ModSAF during large-scale DIS exercises such as STOW-E and Warfighter 95, the system slows down significantly processing network entities. Additional tests are required to determine how the network traffic reduces the number of air entities ModSAF can simulate. In conjunction with these tests, it is also necessary to identify the update rate where the simulation becomes unacceptable for interaction with manned flight simulators.

ModSAF is capable of distributing the processing load across workstations by using the Persistent Object (PO) Protocol. In addition to the DIS protocol, ModSAF workstations on the same PO database number are able to share simulation tasks across the same physical network used for DIS traffic. The system is also able to pick up the simulation tasks of another workstation should it fail. One drawback to the PO protocol is the additional traffic generated on the network. Although the PO protocol uses a different User Datagram Protocol (UDP) port number than the DIS traffic, the traffic is broadcast to all machines participating on the network. This broadcast usually does not create problems for Local Area Networks. It is, however, a consideration in Wide Area Networks because gateways, encryption devices, and the leased commercial lines restrict the total bandwidth available for DIS exercises.

6.0 Results

This evaluation indicates that the current version of ModSAF cannot provide air entities capable of effectively interoperating with human-in-the-loop flight simulators. Although the evaluation focused primarily on the F-16D, analysis of the general fixed-wing aircraft software suggests the same general problems with all fixed-wing aircraft entities.

In general, the software architecture is sound and well organized by the ModSAF developers. Unfortunately the good software architecture forces dependencies on the common fixed-wing aircraft libraries that are incorrect and poorly documented. Therefore, incorporating a correctly modeled air entity cannot be accomplished without first redesigning the existing fixed-wing aircraft, hulls, and other code discussed in Section 4.0. The problems are too big to be corrected with a few patches.

It appears that the air entity simulations were developed without sufficient aid from aerospace or mechanical engineers. As a result, incorrect algorithms were used in some cases to model the aerodynamics and flight mechanics of air entities. In addition, failure to use standard aerospace and simulation engineering conventions has resulted in models that are inefficient and difficult to maintain.

In addition, the following list suggests that the ModSAF F-16 simulation was developed without appropriate subject matter expertise:

- F-16D simulation with no F-16A or F-16C modeled
- No F-16 air-to-air capability
- No two/four ship air-to-air capability for entities with an air-to-air mission
- Limited/incorrect weapon selections
- Threat detection routines do not consider RWR
- Threat detection routines do not consider visual detection for air-to-air missions
- No command and control at any level modeled for air entities
- No threat avoidance capability, air entities blindly follow route

The Graphical User Interface provides an easy interface for creating entities and setting up basic routes. The user's manual is difficult to follow while assigning tasks. All tasks are grouped under fixed-wing aircraft task frames, but not all aircraft are capable of correctly carrying out all tasks. Some can be assigned only to aircraft with a visual sensor, others only to individual entities, not to units.

No documentation of the fixed-wing aircraft configuration menu could be found. The majority of the parameters in this menu should not be available to the operator for modification. Lift coefficient, thrust map, take-off speed, max AOA, etc. should be calculated in the flight equations based on the aircraft's aerodynamic properties and the stores selected by the operator.

7.0 Recommendations

To use ModSAF as the CGF system to depict air entities that will be interacting with human-in-the-loop fixed-wing aircraft simulators during ADS exercises, the following recommendations are provided:

1) Completely redesign the fixed-wing aircraft aerodynamic and flight dynamic algorithms and data files using engineers with backgrounds in these subjects.

2) Incorporate air-to-air and air-to-ground sensor capability.

3) Allow the user to assign the appropriate mission to the aircraft based on scenario requirements.

4) Provide fixed-wing aircraft threat detection, and IFF capabilities consistent with actual aircraft capabilities.

5) Add the capability for air-to-air tasks to be conducted as a two- or four-ship unit.

6) Add a Headquarters or AWACS type command and control logic for both blue and opposition air forces.

AIAA-96-3541-CP

DISTRIBUTED INTERACTIVE SIMULATION OVER FIBRE CHANNEL

George Valentino, Pawel Malinowski, Leszek Wronski, Tom Bohman
SYSTRAN Corporation
Dayton, Ohio

Keywords
Fibre Channel, Fabric, DIS, High Bandwidth, Low Latency

Abstract

The future for Distributive Interactive Simulation (DIS) exercises seems certain. DIS exercises will continue to grow in frequency, complexity, and fidelity. The interaction among simulated entities, already complex, will become more so, as will also the interaction among the various network LAN/WAN segments. Bandwidth, already a critical design constraint for exercises and a major driving force behind current and proposed DIS architectures will become even more of a bottleneck. Low latency will continue to be important, but in the very large exercises harder to guarantee. LAN/WAN segments will evolve from current technologies to new emerging technologies which will alleviate some of these problems. One emerging LAN technology which deserves consideration for its potential contribution to DIS is the gigabit Fibre Channel technology. This paper will first introduce the fundamentals of Fibre Channel technology. Secondly, the paper will map DIS requirements[1] to Fibre Channel in the LAN/MAN environment. Finally, the paper will discuss some DIS/LAN related issues such as WAN and ATM connectivity and Fibre Channel product availability.

Fibre Channel, an ANSI Standard

In 1988 the ANSI sanctioned X3T11 committee was formed to develop an ultra high-speed transport for mass storage, peripheral I/O, and networking communications. The X3T11 Fibre Channel committee had a vision that next generation LAN/MAN and peripheral I/O communication requirements could be handled with a serial gigabit data stream through a single physical connector. To realize this vision the committee borrowed the best elements of I/O channel technology and merged them with the best elements of networking technology. The objective was to develop a serial communication link which supports the bandwidth and reliability needed by I/O channels and the flexibility, connectivity, and distance of networking technologies. The result is "Fibre Channel", a universal carrier or transporter of data, capable of simultaneously handling networking and I/O channel protocols.

Now that the first Fibre Channel standard, FC-PH (Fibre Channel Physical and Signaling Interface) Rev. 4.3, has been approved by ANSI as a standard[2], a group of interested companies have joined themselves together in association to cooperatively work toward Fibre Channel acceptance into the commercial market. The Fibre Channel Association (FCA) really has two major objectives. First, it works with the ANSI X3T11 committee's standards to further define interoperability profiles. For instance, one such profile, the IP, defines how one should implement their FC-4 mapping to IP for the sake of interoperability with others running IP on Fibre Channel. Secondly, the FCA members conduct joint market ventures raising public awareness of Fibre Channel as a next generation technology.

Two market studies have been conducted which focus on Fibre Channel's market potential. In 1993 only two million dollars in total Fibre Channel business was transacted. But 1995 was on target with over fifty million dollars in business conducted. By the year 2000 the Fibre Channel market will swell to 2.5 billion dollars and by 2003 it will be in excess of four billion dollars. There are now over 100 companies developing Fibre Channel products. The companies which comprise the FCA include silicon vendors, board level suppliers, and system suppliers. SYSTRAN Corp. is a FCA principal member, and member of the Board of Directors, and is working with the other members to define the standards for FC products and systems.

Fibre Channel technology can support multiple requirements in a distributed communications and computational system, as illustrated in Figure 1.

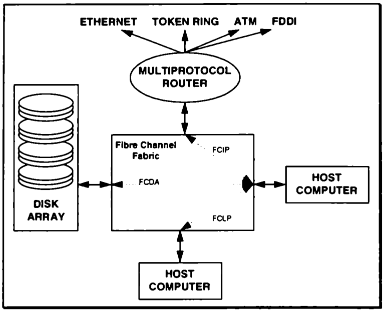


Figure 1 Breadth of Fibre Channel Applications

Fibre Channel Technical Overview

Fibre Channel is flexible, powerful, and extendible without undue complexity. This section of the paper will review at a high level the basic concepts underlying the technology. Namely, the layers and baud rates, topologies, and classes of service offered.

Layers and baud rates

Fibre Channel defines several implementation layers. See Figure 1. The FC-0 physical layer defines such things as the supported baud rates, media, and connectors. Currently supported serial baud rates range from 133 Mbits/sec, 266 Mbits/sec, 531 Mbits/sec, to 1.0625 Gigabits/sec. Additionally, work has begun for a follow-on standard to define 2 and 4 gigabit serial baud rates. In spite of the name "Fibre Channel", FC-0 supports both optic and copper media to distances up to 10KM.

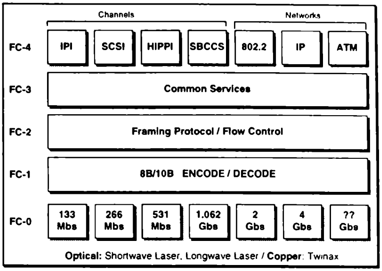


Figure 2 Fibre Channel Implementation Layers

The next layer, FC-1, defines the 8B/10B transmission protocol of the data. This is an adaptive code designed to provide data reliability, DC balance, and word alignment. The FC-2 layer is the transport service layer for Fibre Channel. Data recognized by FC-3 is transparent to FC-2. FC-2 defines the basic signaling and framing protocol including sequence management, addressing, flow control mechanisms, and topology information. FC-4 defines the mapping of various networking and I/O channel protocols to Fibre Channel. One advantage of this flexibility is in the reuse of existing software drivers.

Fibre Channel may be viewed as a high-performance bi-directional connection between any two points. This fact is illustrated in Figure 3.

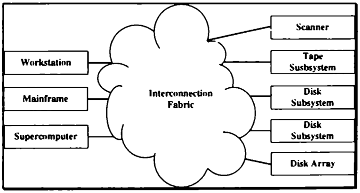


Figure 3 Fibre Channel Provides High-Performance Interconnects

The Fibre Channel interconnection "Fabric" or "Cloud" in a generic manner connects nodes, known as "N_Ports", which are as diverse as disk subsystems, tape subsystems, scanners, Workstations, Mainframe computers, and Supercomputers. The Fibre Channel standard defines how an N_Port communicates with a Fabric and another N_Port, but not how the Fabric performs and manages the connection. In this way Fibre Channel switching or connection Fabrics may be developed to specific applications making use of the latest technology and the most appropriate topology and not be burdened with an outdated standard force fitting old technology into new evolving problems.

Topologies

The "Cloud" of Figure 3 may configure N_Ports or connections to the network in any combination of the three topologies depicted in Figure 3. Point-to-Point, Switched, and/or Arbitrated Loop[3] topologies allow the system integrator a great deal of architectural flexibility. The system integrator can mix/match the best architecture for the application. For instance the data bandwidth of a gigabit Arbitrated Loop is 100 megabytes. This is plenty of bandwidth for many applications and alleviates the extra cost of a switch. It also provides for very simple LAN implementation due to the shared media and broadcast applications.

Communication using Fibre Channel is conducted using a construct called an "exchange." An exchange, in turn, is composed of one or more nonconcurring "sequences" for a single operation, with each sequence composed of a series of "frames." This organization is illustrated in the left-side of Figure 5.

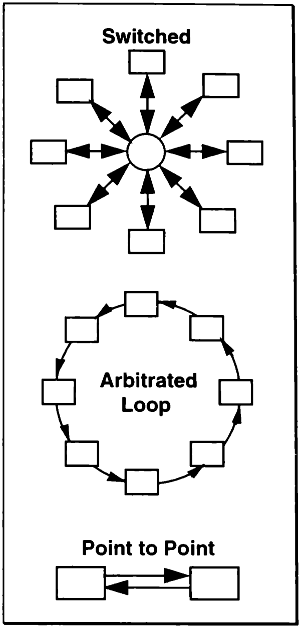


Figure 4 Fibre Channel Topologies

Each frame is composed of:

- a start of frame (SOF) indicator (4 Bytes)
- a frame Header (up to 24 Bytes)
- the Payload (2112 Bytes) - this can include an Optional Header of 64 Bytes (limiting the Payload to 2048 Bytes)
- error checking using cyclic redundancy check (CRC) (4 Bytes) and,
- the end of frame (EOF) indicator (4 Bytes)

This frame composition is illustrated in the top portion of Figure 5, with details of the frame Header illustrated in the center of the figure.

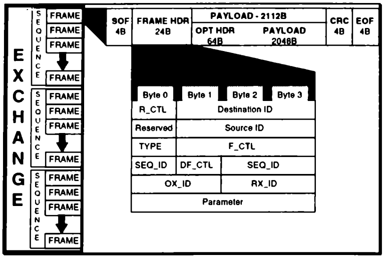


Figure 5 FC-2 Framing and Protocol

Fibre Channel supports five classes of services as highlighted in Figure 6. Class 1 (circuit switch) is a circuit switched operation which guarantees bandwidth and delivery of data. Once a connection is made through a Fabric it is maintained until broken by one of the connected N_Ports. Class 1 also offers in-order delivery of data because the connection is maintained and not broken or changed. Latency is minimal because once a connection is made the packet headers are not looked at for destination addressing.

Class 2 (frame switch) is a packet (frame) switched service where packets may be switched on packet frame boundaries within the Fabric. Flow control and data acknowledgments are implemented guaranteeing delivery of the data. But in-order delivery of data is not a given, unless the Fabric happens to be an Arbitrated Loop where only one path for data flow exists. Also bandwidth is not guaranteed even though the flow control mechanisms will deliver the data eventually.

Fibre Channel Class 3 (datagram) service is a datagram packet switched service with no guarantees of delivery. Packets may be dropped by the Fabric (unless AL) if collisions occur and because there are no acknowledgments, hence, the packet will not be resent.

Fibre Channel Class 4 (virtual circuit) is a fractional connection oriented service which allows a portion of the bandwidth of a connection to an N_Port to be allocated and guaranteed. An N_Port may have up to 254 Virtual Circuits maintained with Quality of Service parameters which include guaranteed bandwidth and latency.

Fibre Channel Class 5 (isochronous) is a fabric assisted, connection oriented service that utilizes permanent and dynamic virtual circuits (VCs), offers quality of service (QoS) per each VC, guarantees maximum bandwidth, delay, is jitter and congestion-free, provides 65,536 VCs per port, and uses TDM to multiplex VC's allowing different data rates at sources and destinations.

- Class 1 - Circuit Switch
    - Guaranteed Bandwidth and Delivery
    - Connection Service
    - End-to-End Flow Control
    - In-Order Delivery
    - Circuit Switched
- Class 2 - Frame Switch
    - Guaranteed Delivery
    - Connectionless Service
    - Buffer-to-Buffer Flow Control
    - Packet Switched
- Class 3 - Datagram
    - Best Effort Delivery
    - Broadcast
    - Datagrams
    - Packet Switched
- Class 4 - Virtual Circuit
    - Fabric-assisted, Connection service
    - Guaranteed Delivery
    - Virtual Connections Between Ports
    - Guaranteed Quality of Service
- Class 5 - Isochronous
    - Fabric-assisted, Connection service
    - Permanent and dynamic VCs
    - Quality of service per VC
    - Guarantees maximum bandwidth
    - Guarantees delay
    - Different data rates at sources and sinks

Figure 6 Fibre Channel Classes of Service

It is a major advantage for Fibre Channel that a single Switching Fabric can handle multiple classes of service. Typical network traffic is characterized in this way; 80% of the messages are small single frame sequence packets and 80%

of the bandwidth is consumed by 20% of the messages, which are large multi-frame sequences. It is very difficult for a one dimensional switch to efficiently handle both types of messages. The large bandwidth messages require a large allocation of buffering resources since they inherently cause worst case traffic congestion for a packet switch. Collision and packet dropping is likely (only in Class 3) because the chance of a small message needing one of the internal packet switch buffers used by a large message is very high.

A two dimensional switch solves the traffic congestion problem. The 20% messages which are taking all the bandwidth are generally handled most efficiently with a class 1 circuit switched service. Very little buffering is needed and latency is reduced in that once the source to destination path is locked, no effort needs to be expended on routing. The small messages which account for 80% of the total number of messages are handled best by the class 2 or class 3 service. There must be adequate buffering to allow the packet frame header to be decoded for destination information and to allow a source to destination path to be located. Collisions may occur necessitating buffers on internal switch paths, on the input stream or on the output stream. A Fibre Channel two dimensional switch efficiently handles both of these diverse traffic patterns. The class of service identified in the packet Start-of-Frame primitive routes the Class 1 or Class 2 traffic to the appropriate switching mechanism within the Fabric.

Perspectives for Fibre Channel

Fibre Channel offers flexibility, capacity, and speed for applications in a number of domains. Figure 7 shows the bandwidth requirements for various activities (e.g., file server backbone, compute servers, etc.) for different bus interfaces. Fibre Channel gigabit network technology is ideally suited for these applications.

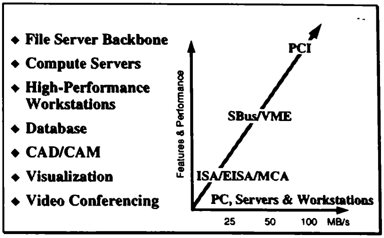


Figure 7 Gigabit Networks

Storage systems can also benefit from Fibre Channel technology. Figure 8 illustrates various application sectors (e.g., data warehouses, decision support systems, etc.) and the high-bandwidth operating-points where fibre channel solutions make sense.

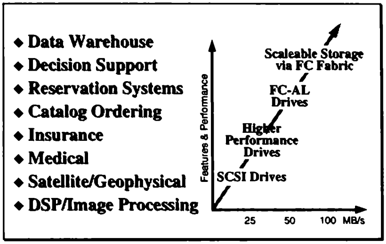


Figure 8 Next Generation Storage

Finally, Figure 9 shows examples of next generation compute clusters (e.g., PCI/PMC-based high-end computers) and the applications domains where Fibre Channel transport technology can bring substantial benefits (e.g., radar and sonar systems).

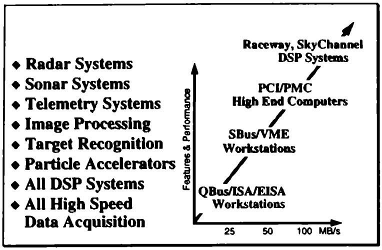


Figure 9 Next Generation Compute Clusters

Mapping DIS Requirements to Fibre Channel

Current Requirements

Figure 10 lists current and future DIS LAN requirements. As indicated in the figure, Fibre Channel is more than adequate to meet the stringent requirements imposed by DIS. In fact, Fibre Channel meets these critical requirements, including bandwidth and latency, better than any other existing or emerging technology.

DIS Current and Future LAN Requirements

- Open Standard
- Bandwidth
- Latency
- UDP
- TCP
- IP
- Broadcast
- Multicast
- Permanent Virtual Circuits
- Switched Virtual Circuits
- Reliable Data Delivery
- Best Effort Data Delivery
- WAN/ATM Connectivity

Figure 10 DIS LAN Requirements

Fibre Channel has a great advantage over other networking technologies simply because of the speed of the serial links. There are current implementations of Fibre Channel for four baud rates; 1) 133 Mbits/sec, 2) 266 Mbits/sec, 3) 531 Mbits/sec, and 4) 1.0625 Gigabits/sec. After protocol overheads are deducted these baud rates convert to 12.5 Mbytes/sec, 25 Mbytes/sec, 50 Mbytes/sec, and 100 Mbytes/sec data rates respectively. No commercial standard network known has as high a serial bandwidth. As a high-speed backbone Fibre Channel is unparalleled in performance. Existing implementations of the Arbitrated Loop topology easily handle 100,000+ packets per second. The ultra high speed serial links also translate into very low latencies. Typical buffer-to-buffer latencies in Fibre Channel implementations are in the sub-millisecond range. IP stack processing would be added to this on each end to obtain Application-to-Application time.

Fibre Channel can also make significant improvements over current DIS architectures in its mapping to UDP. First of all, compatibility with current DIS application software is maintained, because Fibre Channel provides transport for UDP/IP protocols through a mapping layer called FC-4 and FC-4 encapsulates IP packets into LLC/SNAP 8802 packets. The improvement is seen in that Fibre Channel computes the checksum automatically in hardware. The IP and UDP stacks could be improved by reducing message latency. This is accomplished by eliminating the checksum in the IP stack.

Fibre Channel's support for Broadcast and Multicast services is another improvement that could benefit DIS. NL_Port devices (still being defined) which support the Arbitrated Loop topology may broadcast to a well known address (0xffffff). N_Port devices compliant with the Fibre Channel FC-PH2 standard support broadcast in Fabric topologies (packets are being duplicated inside the Fabric to each output port). FC-PH2 compliant devices may also multicast through a Multicast (Alias) Server, which maintains all information about multicast groups, connected to the network.

Another Fibre Channel benefit to DIS has to do with the wide spectrum of communication models offered and supported by Fibre Channel. DIS

PDU's could be sent in datagrams (Class 3), acknowledged packets (Class 2), through a circuit (simplex or duplex - (Class 1) , or virtual circuit (Class 4), fulfilling and extending DIS requirements for the communication model. One important feature which Fibre Channel is offering to existing DIS applications is reliable delivery without the need for implementing any Reliable Delivery Protocols on top of UDP. All the Fibre Channel acknowledgment frames are generated in hardware.

Fibre Channel provides an extremely cost effective LAN backbone solution. The 100 Megabytes/second bandwidth of a gigabit link and the Arbitrated Loop topology which does not require a switch makes this possible. Concentrators for Arbitrated Loops are now available which automatically configure all connections into a Loop and bypass any port not active.

Future Requirements

Fibre Channel will also meet future requirements of DIS. Sometime in the future the prevailing WAN for DIS exercises is likely to be ATM. DIS will want to take advantage of the Permanent and Switched Virtual Connections that ATM offers. Fibre Channel FC-PH2 compliant devices will also support Permanent and Switched Virtual Connections (class 4 service) with setup/remove and activate/deactivate phases.

Once again, Fibre Channel not only meets this requirement but also offers an added value above other technology choices for the LAN. The Fibre Channel advantage is that not only is bandwidth guaranteed but the delivery of data within virtual connections is also guaranteed. The FC-PH2 standard accomplishes this by a traffic management protocol which guarantees the availability of the requested bandwidth. Network layer resource reservation is also supported to establish and maintain resource allocations across networks that assure an adequate Quality of Service by a QoS Facilitator/Agent.

An unique benefit of Fibre Channel for DIS applications is the potential for mapping DIS Protocol (PDU's) directly to Fibre Channel Layers without any intermediate protocol stack. This would reduce processing time on each host and better accommodate the very high traffic generated by DIS exercises. There is already an effort within the Fibre Channel Committee to define UDP directly over Fibre Channel. Reliable UDP can be performed directly over the Fibre Channel Link Encapsulation Layer (FC-LE standard)[4] by using Class 1 and Class 2 connections. The advantage of using FC-LE is that packets could be routed to other LANs such as Ethernet or FDDI. In a pure Fibre Channel environment, FC-LE is not necessary. A separate implementation of UDP on top of FC-LE could coexist with the more general UDP/IP/ FC-4 (LLC 8802) implementations in any system.

LAN Emulation

ATM is designed primarily for the WAN environment. LAN Emulation is designed to help ATM look and act like a LAN. However, LAN Emulation on ATM is not very efficient. Servers are required to handle the Address Resolution Protocol (ARP), Broadcast, and LAN Configuration. On the other hand Fibre Channel is a Local Area Network itself and no LAN emulation is required. A Fibre Channel LAN client working in an Arbitrated Loop topology is similar to Ethernet and FDDI in that there is no additional protocol burden to connect to the network, so no LAN Configuration Server is required. Fibre Channel handles broadcast natively so no Broadcast Server is required and the ARP protocol runs elegantly. Also any Fibre Channel N_Port may be configured as an ARP client or as an ARP server.

Wide Area Network Connections (Tunneling Fibre Channel Frames Over ATM)

In order to be successful as a DIS LAN, Fibre Channel networks must be able to interoperate with ATM WANs. Although many bridging, mapping, and tunneling solutions will become available, the earliest candidate solution will be tunneling.

Figure 11 illustrates the mapping of a DIS information unit (PDU) to Fibre Channel which is tunneled over ATM. One IP packet (max payload 65496 Bytes) would be fragmented into a sequence of Fibre Channel frames. Then each of the Fibre Channel frames (max payload of 2112 Bytes) would pass through an ATM Adaptation Layer (AAL5) and be fragmented into ATM cells.

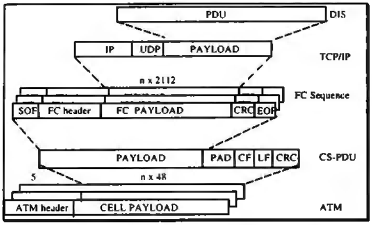


Figure 11 Mapping DIS PDUs to FC to ATM

Another more elegant solution for Fibre Channel ATM interoperability has been defined by the X3T11 committee. The port in a Fibre Channel Fabric (Switch) responsible for interconnecting with other Fabrics or Wide Area Networks is called an E_PORT. An E_PORT will support a flow of Fibre Channel transmission words through the ATM network to another compatible E_PORT in another switch element. This is achieved by transforming Fibre Channel frames into ATM adaptation Layer 5 (AAL 5) CPCS-PDUs and back. It is a function of a module called an ATM Inter-Element Link Translator (ATM IEL) defined by the FC-PH standard (Annex C) and implemented inside the E_PORT. The IEL is used to carry Fibre Channel frames, primitive signals, and primitive sequences between switch elements and acts as a bridge between distantly located switch elements. Exact translation between Fibre Channel signals, primitives and ordered sets is included in the FC-PH standard.

Figure 12 serves as a reference model for Fibre Channel to ATM connectivity. Applications running correctly on a closed Fibre Channel network will continue to operate correctly when operating across an ATM network. Performance differences may arise because of the lower bandwidth and greater distances but operationally the application programs should require no changes.

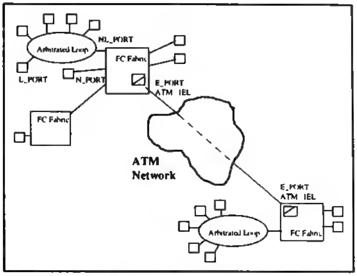


Figure 12 Reference Model for FC to ATM Connectivity

Fibre Channel Product Availability

First generation Fibre Channel N_Port host adapters and Fabric switches will be in full production by first quarter '96'. Many companies, like SYSTRAN Corporation, are shipping production quantities of N_Port host adapter cards to end-users today. SYSTRAN Corporation has host adapter N_Port cards which support Class 1, Class 2, and Class 3, as well as Switched, Point-to-Point, and Arbitrated Loop topologies at all Fibre Channel speeds. These N_Port host adapter cards are available for VME32/64, Sbus, PCI and PMC. Figure 13 illustrates a 6U-VME32/64 FibreXpress N_Port adapter.

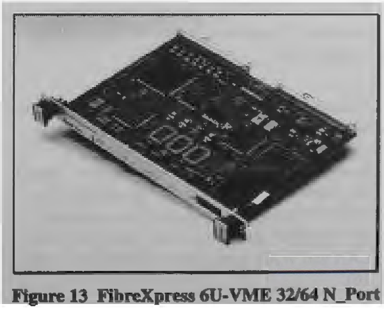


Figure 13 FibreXpress 6U-VME 32/64 N_Port

Figure 14 illustrates a PCI FibreXpress N_Port adapter.

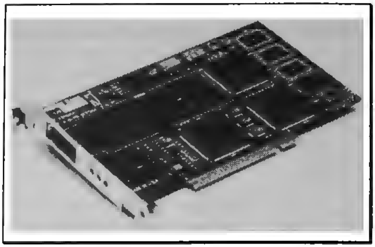


Figure 14 FibreXpress PCI N_Port

In addition, we have a 32 port Transparent Switch in Beta design debug. This Transparent Switch, notionally illustrated in Figure 15, is composed of a high-speed, high-bandwidth crosspoint switching fabric plus a series of N_Port interface modules.

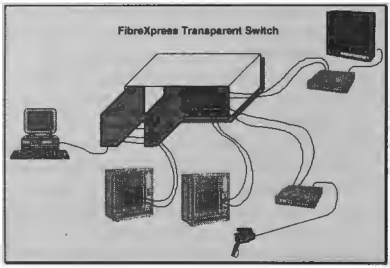


Figure 15 The Transparent Switch Can Interface Numerous Devices at Gigabit/Sec Bandwidths

Using a Fibre Channel switch like the Transparent Switch, plus a series of Fibre Channel N_Ports, allows the real-time user to configure high-bandwidth networks for conducting live, virtual, and constructive simulations. In addition, the same technologies can support the real-time movement of data, voice, and video for other LAN and WAN applications (illustrated in Figure 16).

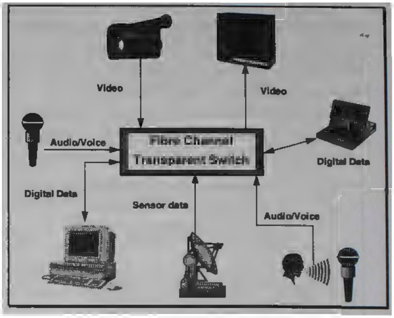


Figure 16 Fibre Channel switches can support data, voice, and video in LAN and WAN applications.

Additional information regarding FibreXpress N_Ports and other Fibre Channel products can be obtained by contacting SYSTRAN Corporation in Dayton, Ohio (513) 252-5601 or at World Wide Web site *http://www.systran.com*.

Summary

In summary, Fibre Channel offers multiple advantages to the DIS community, and to any user that must utilize distributed computational assets. Figure 17 lists the overriding advantages of Fibre Channel.

- ◆ ***ANSI Standard*** **from the Computer Industry**
- ◆ ***Removes*** **Today's Limitations**
- ◆ ***Delivers*** **Speed and Reliability**
- ◆ ***Provides*** **Distance, Connectivity, and Low-Cost**
- ◆ ***Guarantees*** **Bandwidth and Delivery**
- ◆ ***Gives Flexibility*** **of Any Protocol or Topology**

Figure 17 Fibre Channel Advantages

The Fibre Channel Association

Additional information about the Fibre Channel protocol, standards, working groups, and related issues can be obtained from the Fibre Channel

Association (FCA), 12407 Mopac Expressway North 100-357, P.O. Box 9700, Austin, TX, 78766-9700. Phone (800) 272-4618. The FCA site on the world wide web is at h*ttp://www. amdahl.com/ext/CARP/FCA/FCA.html.*

Author Information

George J. Valentino is a Senior Technical Manager and leader of SYSTRAN's Innovative Systems Group. He can be reached at *gvalentino@systran.com.*

Pawel Malinowski, is the Senior Software Engineer responsible for developing SYSTRAN's Fibre Channel device drivers (i.e., Lightweight Protocol, TCP/IP) and embeded software. He can be reached at *pmalinowsk@systran.com.*

Leszek Wronski, Ph.D., is SYSTRAN's FibreXpress IR&D Manager. He can be reached at *LWronski@systran.com.*

Tom Bohman is the Technical Marketing Manager at SYSTRAN. He can be reached at *TBohman@systran.com.*

Bibliography

[1] 'Communication Architecture for Distributed Interactive Simulation (CADIS),' IST-CR-93-13, June 1993.

[2] Fibre Channel Physical and Signaling Interface (FC-PH) Revision 4.3, August 24,1994

[3] Fibre Channel Arbitrated Loop (FC-AL) Rev 4.5, June 1, 1995

[4] Fibre Channel Link Encapsulation (FC-LE) Rev 0.92, August 7, 1995

FibreXpress is a trademark of SYSTRAN Corporation.

AIAA-96-3542-CP

TECHNIQUES FOR REDUCING LATENCY IN DISTRIBUTED FLIGHT SIMULATION

George J. Valentino, Stephan A. Lubbers, Stephen T. Thompson,
Kenn W. Scribner, Kevin E. Breeding

SYSTRAN Corporation
Dayton, Ohio

Keywords
Latency, DIS, Replicated -Shared Memory, SCRAMNet®, Phoenix, SNAP, SNAPpy, software agents

ABSTRACT

Latency. Measure it, account for it, and reduce it! Latency is relative. A 100 millisecond closed-loop latency in a PITL (Person-In-The-Loop) tank system or simulator may be acceptable. A 300 millisecond to 5 second closed-loop latency to a CITL (Commander-In-the-Loop) C^3I system or simulator may also be acceptable. Such latencies would not be acceptable to a simulated flight of four F-16s during a training ingress to a hostile situation. The CADIS (Communications Architecture for Distributed Interactive Simulation) uses the OSI (Open Systems Interconnection) reference model as a means to illustrate and specify acceptable latencies. The seven layers in this OSI reference model include the top-level application layer, followed by the presentation, session, transport, network, and data link layers, with the physical layer at the bottom. For a tightly coupled system (such as flight simulations), CADIS allows up to 100 milliseconds from the application-layer at one node, down the ISO-stack, across the network to the next node, and up this node's "stack" back to the second application layer. In addition, CADIS permits only 10 milliseconds for the transfer of data either from an application layer to the physical (or physical layer to application layer). For two nodes, this "stack" time would be 20 milliseconds, maximum. This leaves just 80 milliseconds as the total allowable 'intersystem' latency. At SYSTRAN, we are working on latency reduction techniques across two fronts: (1) by the architecture of the network and (2) through the use of multi-processor NIU (Network Interface Unit) techniques to prevent the introduction of this latency in the first place. This paper will address these topics: (1) alternate network architectures that support low-latency, distributed simulation; (2) specific replicated shared-memory devices that aid in implementing these architectures; (3) the use of a Windows NT-based multiprocessor system for latency reduction within the network interface unit; and (4) related topics and devices.

Latency Will Ruin Your Simulation

Consider a hypothetical scenario: a simulated few-on-few air combat scenario that includes high-performance friendly and aggressor aircraft. Nothing has been spared in setting-up this simulation - a "no holds barred" grudge match between the Blueberry, WI, Friendlies and the Red Lion, PA, Aggressors. Each simulated aircraft cockpit is outfitted with control systems equipped with force-feedback systems, sunlight-readable multi-function displays, an actual flight-worthy HUD that has been converted for use in the simulator, and replica ACES-II seats. In addition, each aircraft cockpit is placed within the envelope of a dome projection screen, which is illuminated with a simulation-driven horizon, simulation-driven clouds, and high-resolution laser projectors (for projecting other aircraft).

The simulation facility has recently been upgraded to be DIS-compliant, with each simulated aircraft being interfaced via a NIU to the DSI (Defense Simulation Internet).

Several person-years of activity were expended to prepare for a major demonstration of a new laser weapon system. All Blue and Red pilots have been thoroughly trained in flying their aircraft and in using the available weapons. Prior to this demonstration, all training was done in local mode, off the net.

After all the Generals have been introduced, the demonstration begins. At t = 0:18:15, Blue3 is attempting to lock onto Red2. The Blue3 pilot, who was able to lock onto any target in training, cannot lock on to Red2. While attempting to re-calibrate his sensors and use the new laser weapon, he is destroyed by a simulated missile fired by Red2. From the gallery, General I. U. Seada, his mouth twisted into an unnatural form, was heard to utter 'What happened...it worked on paper?'

'It was the latency, sir!'

During post-demonstration analysis, it was discovered that the Blue3 pilot would never have been able to lock onto Red2. The latency between when the state of Red2 was computed (in the simulation computer at Red Lion, PA) and when the new position of Red2 was generated (in the dome at Blueberry, WI) was simply too long. This difference, notionally illustrated in Figure 1, caused the demise of Blue3. 'P' represents the perceived position of Red2 as sensed by the Blue3 pilot and the computers at the Blueberry simulation center. 'C' represents the position of Red2 as computed at the remote Red Lion simulation center.

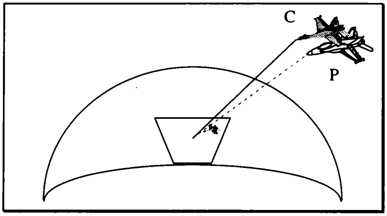


Figure 1 Mismanaged Latency Will Effect Simulation Fidelity

The CADIS Standard for Latency

The DIS standard, protocols, and reckoning algorithms represent a collection of well constructed message-passing communication and processing algorithms and techniques that will meet or exceed the needs of many DIS "players" in many application domains. For those cases where 100ms latency and/or appropriately "reckoned" updates will satisfy the needs of the simulation, we heartily endorse its use. When 100ms latency is too long, other approaches must be utilized.

Latency Margins

Figure 2 shows an ISO model with CADIS stipulated latencies[1]. For a tightly coupled system, up to 10 ms is allocated to the movement of data from the application layer down to the physical layer. Going back up again (at the other node), results in a total "stack time" of 20 ms. This leaves just 80 ms as the total allowable 'intersystem time' latency.

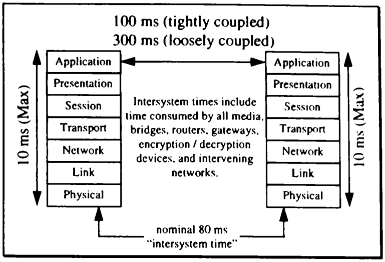


Figure 2 CADIS Standard Latency Values

Latency has always been the key issue in person- and hardware-in-the loop simulation. The 100ms and 300ms closed-loop response times stipulated by the DIS Standard may not be sufficient to meet other simulation requirements. In particular, this criteria falls short for either of these two cases:

- when the vehicles are highly dynamic (combat aircraft, not tanks or trucks), or,

- when the players are within visual range and are interactive (e.g., dismounted infantry moving together in the same virtual world).

Measuring Latency: It's a SNAP

A scenario very much like the one depicted above occurred during an Advanced Research Projects Agency simulation, the record-setting War Breaker "Zen Regard" simulation.[2] At several times during the simulation, participants were actively engaged in air combat in highly-dynamic fighter aircraft. Frequently, "sure hit" missiles "lost lock" and failed to destroy the intended target. At other times, aircraft on the network "disappeared", only to reappear in odd locations (at least locations not predicted by the locally-held dead-reckoning algorithms). Air combat over the network was a dismal failure, at least from a training and data collection perspective.

From that failure, though, one of the participants, the Control Integration and Assessment Branch, Flight Control Division, Flight Dynamics Directorate, Wright Laboratory, Wright-Patterson Air Force Base, realized latency was the serious issue in realistic distributed simulation. Latency caused the missile failures, as the computationally intensive missile flight algorithms were fed erroneous target positional information. Latency caused target "jumping" or "warping", which not only confused the computers but also irritated the highly motivated and aggressive pilots. Latency also caused network "timeouts", which resulted in participants "dropping out of" the simulation until their network packets were again sent over the network.

Two specific questions were asked:

- Where in the simulation were the individual elements of latency introduced?
- What could be done to measure the latencies involved in the overall simulation?

To this end, engineers at the Control Integration and Assessment Branch developed the Simulation Network Analysis Project (SNAP) computer system[*]. SNAP's main function is to stamp accurate time information on simulation "events". SNAP's original design included not only network events, such as the receipt of a Protocol Data Unit[Ψ], but also electrical events, such as analog and digital inputs to the SNAP computer (from the simulator), and image generator events, such as horizon roll and pitch (through custom hardware the Control Integration and Assessment Branch designed and produced).

A notional view of SNAP is provided in Figure 3.

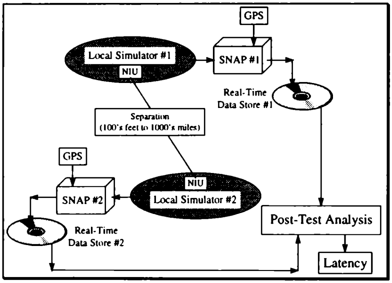


Figure 3 SNAP Top-Level Diagram

SNAP would determine accurate event time by initializing a local clock using time data received from the Global Positioning System (GPS) satellite constellation. To reduce slew errors in the local clock, the clock's oscillator was encased in an oven-controlled enclosure. To reduce the effects of clock slew rates between SNAP computer systems[ξ], each system's local clock would periodically be re-adjusted (off-line) using re-acquired GPS time data.

Physically, SNAP consisted of ruggedized personal computer systems using 50MHz Intel i80486 processors on International Standard Architecture (ISA) bus motherboards. So that measurements taken by the system would be deterministic, the core SNAP software was written in the 'C' programming language and used the Intel iRMX operating system, which is a real-time, preemptive multitasking operating system. The SNAP user interface was written for the MS-DOS environment using National Instrument's "LabWindows" graphical user interface. After system initialization, MS-DOS ran as a "task" under iRMX, giving the system a flexible user interface on top of a deterministic real-time operating system. Using the SNAP system, simulation engineers could now both measure and drive (stimulate) simulator inputs for experimental purposes.

Prior to a simulation, a simulation engineer would connect the SNAP computer system to the local simulation. This would include an Ethernet connection (for Ethernet packet and Protocol Data Unit reception and timing), any desired analog and digital signals (such as from a control stick or switches of interest), and the image generator measurement hardware, known as the Electronic Visual Display Attitude Sensor, or EVDAS[⚕]. EVDAS continually reads the pitch and roll angle from the raster display. It calculates the aircraft player angles during the active raster field and transmits the data to the SNAP computer during the vertical retrace interval. This allows EVDAS to generate an interrupt every single field (i.e., at 60Hz) and outputs the angles every field.

During the simulation, the simulation engineer would monitor SNAP's data collection processes. Periodically, data could be saved to a file for later

[*] Messrs. Ron Ewart, Barry Bryant, Captain Scott Douglass, and Jeff Slutz.

[Ψ] The Protocol Data Unit, or PDU, is the basic unit of information exchange between simulators participating in a Distributed Interactive Simulation. It contains a variety of information regarding the current state of the simulation.

[ξ] At this time there are three SNAP systems, which is why the clocks on each must be synchronized prior to data collection.

[⚕] EVDAS has the capability of vertical, horizontal, and circular color reference measurement, as well as a centroid-tracking feature.

analysis. After the simulation, SNAP data files could be analyzed and compared.

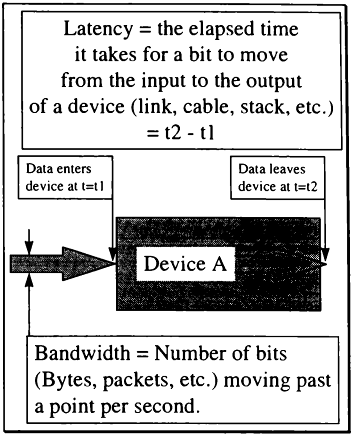
Simple latency calculations involved the subtraction of event time stamps. The difference, or total latency between events, would be accurate to GPS time. In other words, the measured latency data contains a time error of approximately 100 microseconds[ΨΨ] (considered to be quite accurate for the systems under scrutiny). Because events would be generated from a variety of simulator sources, SNAP would produce latency measurements not only for network traffic, but also for processes internal to the simulator. It was now possible to measure the latency involved from pilot input (stick movement), to visual system update (using EVDAS), to Protocol Data Unit generation, to finally the network communication latency (travel time site to site for the Ethernet packet).

Interestingly, the communication latency was not the culprit (this was the pre-experimental hypothesis). SNAP experiments to-date indicate most of the latency is incurred during Protocol Data Unit generation and reception[¶]. That is, the efficiency of the Network Interface Unit (NIU) was directly responsible for most of the (bad) latency incurred in the simulation. As a direct result of this, there are several efforts ongoing to build more efficient NIUs and even to improve the basic protocol efficiency to reduce data conversion time.

SNAP information can be found in Bryant, et. al.[3] and Woodyard and Barnhart[4].

High-Bandwidth Does NOT Mean Low-Latency

It is important to understand that simply increasing the bandwidth capability of a network interface and associated media does not, in and of itself, decrease the latency of the data transfer. Increasing the bandwidth of a PCI interface rated at 132 Mbits/sec to a Fibre Channel interface rated at 1062 Mbits/sec says nothing about the latency of the transfers. As illustrated in Figure 4, capacities do not imply speed of movement.



Latency = the elapsed time it takes for a bit to move from the input to the output of a device (link, cable, stack, etc.) = t2 - t1

Bandwidth = Number of bits (Bytes, packets, etc.) moving past a point per second.

Figure 4 Bandwidth vs. Latency

Reducing Latency #1: Remove the Stack

Before the advent of DIS geographically distributed interactive simulation, various techniques were used to meet the latency reduction needs of localized interactive simulation. One of the most widely used techniques for latency reduction was the use of a replicated, shared memory model as the basis of the simulation.

Replicated shared memory uses complete physical copies of the memory (or that portion of the memory that must be shared among all applications) at each processor node on the network. From a logical point of view, the simulation programmer sees one large memory with many attached processors. As computational demands increase, additional processors can be added to the simulation, with simulation tasks reallocated among the new complement of processors.

The physical implementation of replicated shared memory can be accomplished in several ways. One way is to add additional processors within the same processor cabinet (as is done by most multiprocessor computing platforms). Another way is to allow the simulation designer to utilize a mixture of heterogeneous processors on a local area, high-speed

[ΨΨ] Although the SNAP GPS clock has an accuracy of about 3 microseconds, the SNAP system does not record the time that accurately because of several reasons: it takes a finite time to read the interrupt, service the interrupt, and there is some crystal clock drift. Perhaps some other subtle factors as well. At present, the worst case guaranteed accuracy is less than 500 microseconds with all errors included. In practice, at WL/FIGD the accuracy is down to 100 microseconds.

[¶] There is also latency involved with aircraft control, but this closely paralleled the flight dynamics of the simulated aircraft.

network. This latter method is the basis of the SCRAMNet® (Shared Common RAM Network) that has been used extensively in localized interactive simulation since 1989. With SCRAMNet, a complete physical copy of the memory is located within each computing node on the network.

Replicated shared memory reduces the application-to-application latency because the communication paradigm is memory-based (as contrasted to message-passing based communication systems). A memory-based system removes many layers from the ISO 7-layer communications model. Removing layers in this stack removes latency.

Figure 5 provides a model for computing application - to - application latency between synchronized[ξξ] SCRAMNet equipped nodes. In this case, the application to physical layer latency is the sum of two separate latencies: X and Y.

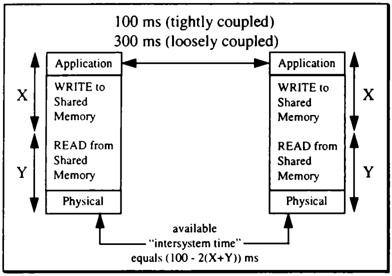


Figure 5 Model For Computing Latency When Using SCRAMNet

Latency X is the time required for an application to WRITE new data to SCRAMNet memory. This is a platform-dependent variable that typically ranges between 500 nanoseconds and 2 microseconds (or 0.0005ms and 0.002ms)[5].

Latency Y is the time required for custom SCRAMNet hardware to READ the data in shared-memory, and to place that data onto the physical fiber optic transmission media. This time consistently ranges between 250 and 800 nanoseconds (or 0.00025ms and 0.0008ms)[5].

The sum of X +Y is the total latency from application to physical layer, within a single node. For SCRAMNet shared memory this sum ranges between 0.00075ms and 0.0028ms.

Using this shared-memory technique, the time now available for 'intersystem time' increases by 19.99ms (100 - 2(0.0028)) to 99.99ms, or almost 25% more margin than that permitted by the CADIS as illustrated above in Figure 2.

For many localized interactive simulations, SCRAMNet has been used as the means to reduce latency and provide a heterogeneous network for the simulation architect.

Reducing Latency #2: Reduce / Modify the Stack

A second way to reduce latency is to either reduce the layers in the stack, or to modify the functions performed in each layer. Removing functionality (and, hence features) from a protocol allows the network architect to arrive at a "lite"[%%] version of the protocol. This "lite" version, a subset of the original, may then be able to achieve the new, lower-latency transfers required by the simulation. Caution needs to be exercised, however, because the new, "lite" protocol may have to scarifice some functions, and may lead to other, unforeseen and perhaps insidious issues in the simulation.

The notion of a "lite" protocol is illustrated in Figure 6.

Reducing Latency #3: Faster Pipes / More Pipes

Another way to reduce latency is to use faster channels and/or more channels. Although this technique may seem to contradict the statements made previously, when properly engineered, they will not. Media access components are now available that can operate at extremely high clock rates, upwards of 1 Gbit/sec for Fibre Channel and SCI (Scalable Coherent Interface) networks.

[ξξ] It is important to synchronize simulations whenever possible to eliminate variable latency on a frame-by-frame basis. SCRAMNet allows simulation architects to include this synchronization feature by firing an interrupt whenever a predefined memory location is updated.

[%%] This "lite" discussion is general in nature and should not be confused with the "DIS Lite" protocol under development by MAK Technologies, Inc.

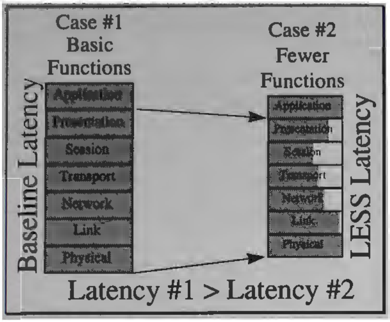


Figure 6 Modify Stack Functions to Reduce Latency

Additionally, multiple channels can be used to further enhance the ability of the system to move data and reduce latency. Research by Kerr and Dobosz indicates the benefit of separating DIS PDUs (Protocol Data Units) "is appreciable. Experimental data shows that there is a clear reduction in the amount of time spent filtering PDUs if there are fewer PDUs to filter."[6]

This concept is illustrated in Figure 7.

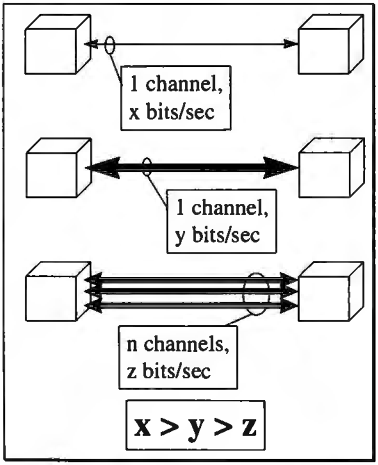


Figure 7 Increase Pipe Capacity / Add More Pipes

Reducing Latency #4: Faster CPUs / More CPUs

Another technique for reducing latency is the use of faster CPUs and/or multiple CPUs in specialized configurations, as notionally illustrated in Figure 8.

The left side of Figure 8 shows a single data stream entering a compute node via a single port, to be serviced by a single CPU. (In this simple example, we also assume that the software application, or process, running on the single CPU is a single-threaded process.)

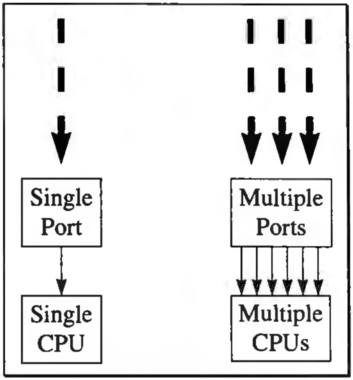


Figure 8 Parallelize Ports and CPUs

This is the classic von Neuman bottleneck that has plagued computer engineers for decades.
The options available to remedy this situation are two: increase the speed of the CPU, or add multiple CPUs.

Increasing the speed of the CPU can provide gains in throughput (and reductions in latency) to a point - then the threshold will be reached again as the complexity of the application increases (and it will). Today the 90 MHz CPU in your desktop PC, that was lightning fast just two years ago, is outdated and cannot meet the requirements of your office. It can be replaced with a new 120MHz or 166MHz processor, but it will still contain the single von Neuman bottleneck.

Given a fixed processor speed, the next option will be to adjust the processing architecture. By changing from a single CPU to multiple CPUs, and by moving to a multi-threaded application, the ability to further reduce latency is enhanced.

At SYSTRAN we are currently incorporating a COTS (Commercial Off The Shelf) SMP (Symmetric Multi-Processor) platform, running Windows NT, as the basis for a multi-threaded DIS network interface unit specifically focused on reducing latency. The SMP platform is a Corollary C-Bus II machine that can accept up to six Intel Pentium processors.[7]

SNAPpy uses Techniques #1 through #4

SYSTRAN engineers are currently six months into an 18-month program to develop and demonstrate a low-latency, DIS (and HLA[#]) compatible, NIU called "SNAPpy." SNAPpy uses the Corollary platform (initially configured with four Intel Pentium 166MHz Processors), multithreaded VR-Link[**] software, SCRAMNet interfaces to simulators in a local cluster, multiple Ethernet ports to other DIS participants, specialized data acquisition and control modules, the SNAP latency measurement technology, and a next-generation EVDAS sensor system.

Reducing Latency #5: Use the Other Machine

Another technique for reducing latency is to execute portions of the application in machines other than your own local machine.

Currently, multiple machines operate in a network and communicate using the conventional method of the Remote Procedure Call (RPC). RPC requires ongoing client-server communication, as illustrated in Figure 9.

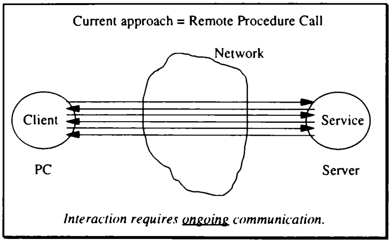

Current approach = Remote Procedure Call
Network
Client
Service
PC
Server
Interaction requires ongoing communication.

Figure 9 Conventional RPC Interaction

This on-going communication consumes bandwidth, and a latency is incurred with each "handshake" of communication.

An alternative to RPC is Remote Programming (RP). In RP, actual code (an application, or an applet) is transmitted as a single bucket-of-bits to another node on the network, where it then executes. This code is called an "agent," or an "intelligent software agent." The RP concept is illustrated in Figure 10.

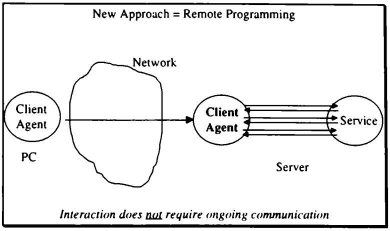

New Approach = Remote Programming
Network
Client Agent
Client Agent
Service
PC
Server
Interaction does not require ongoing communication

Figure 10 Advantage of Remote Programming

Nomenclature surrounding software agent technology is diverse. In addition to the terms "software agent," we also have "intelligent software agent," "mobile cooperative technologies (MCT)," "mobile intelligent networks," and others. All describe the same basic construct -

> "...a new thinking in a design of complex distributed systems as highly organized societies of mobile cooperative intelligent agents which may provide high functionality, openness and robustness."[8]

Additional sources for software agent information include [9,10,11,12].

Reducing Latency #6: Filter, Organize, Prioritize

Swaine and Stapf[13] describe these specific approaches - filtering, organizing, and prioritizing - to managing large simulations that can co-exist with the other techniques described in this paper. These techniques are based upon the Filter Architecture in Figure 11 and include:

[#] HLA is the High Level Architecture, a new initiative by the DIS community to meet the needs of real-time distributed simulation.

[**] VR-Link is a trademark of MAK Technology, Inc.

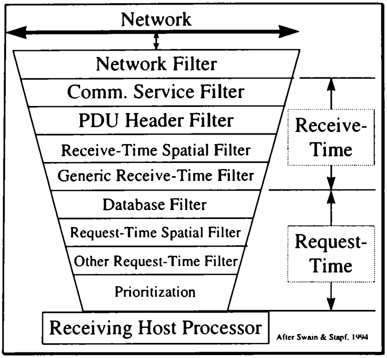


Figure 11 Reducing the Load on the Host Processor

- Network Filtering - "utilizes intelligent routers to prevent data from passing to sections of the network where it is not needed"
- Communication Service Filtering - "eliminate any non-DIS data which shares the physical network media"
- Protocol (or PDU Header) Filtering - "the header of the DIS PDU can be examined for relevance"
- Receive Time Spatial Filtering - "involves the removal of entities that the host simulation could not detect by any means due to the physical geometry between the two"
- Generic Receive-Time Filtering - "perform a series of logical operations on the results of comparisons between user supplied data and fields within incoming PDU"
- Data Organization - "logically group data by PDU type or family so that independent host processes can address certain data types"
- Request-time Spatial Filtering - selectively filter out those entities outside of specific sensor fields-of-view and/or range bins
- Other Request-Time Filtering - "the same generic mechanisms for receive-time filtering can also be employed at request-time"
- Request-Time Prioritization - "If, after all filtering is complete, there is still more data than the host can process, some data must be eliminated"

Low-Latency Data in CLASSIFIED Simulations

The low-latency movement of classified simulation data in distributed simulations poses additional problems for the simulation community. Although the six techniques discussed above still apply, the user is usually significantly constrained by the imposition of security devices and crypto systems never intended for use in real-time simulations.

To meet this challenge, SYSTRAN is mapping the SCRAMNet low-latency protocol to ATM (Asynchronous Transfer Mode) in order to utilize the capabilities of the NSA sponsored, GTE developed FASTLANE ATM encryption system (KG-75)[14].

Wright-Laboratory Requirements

The Flight Dynamics and Avionics Directorates within Wright-Laboratory at Wright-Patterson AFB, OH have used SCRAMNet as an integral part of their individual simulation laboratories. These simulation facilities include aircraft flight control and avionics simulators, simulations, and actual LRUs-in-the-loop. These facilities conduct simulations of 'complete' and 'detailed' individual and multiple high performance aircraft, using update rates of 40Hz, 60Hz, and soon 80Hz (with updates of 25 milliseconds, 16.67 milliseconds, and 12.50 milliseconds respectively).

In order to conduct composite flight control and avionics experiments, both directorates can interconnect these simulations via optical fiber. The length of the optical fibers which connect these two facilities is approximately 10,000 feet.

Unfortunately, only DOD Unclassified communication transfers are authorized over the optical fiber. This is an untenable situation given the requirement to conduct DOD Classified simulations during the course of flight control and avionics research demanded by the R&D mission of these facilities.

SYSTRAN is currently meeting this requirement by developing a bridge that will map SCRAMNet information into ATM packets and vice-versa. We have denoted this device the "Phoenix SCRAMNet-to-ATM Bridge," or "Phoenix" for short.

Figure 12 illustrates how the Phoenix will be used to connect the two SCRAMNet rings in use at the Flight Dynamics (WL/FI) and Avionics (WL/AA) Directorates at Wright Laboratory.

Prior to connecting the two Phoenix and two FASTLANE devices, each simulation network can operate autonomously, thus meeting routine simulation requirements of the representative directorates.

When the Phoenix bridges are added, a low-latency connection will be made between WL/FI and WL/AA, permitting a combined and synchronized, but Unclassified simulation. All data is passed in full-duplex mode via the ATM protocol.

When the two FASTLANE devices are added, then combined and synchronized Classified simulations may be conducted.

Phoenix Meets the Requirements

The goal of the Phoenix product development is twofold: (1) to map data in SCRAMNet memory locations into ATM payloads (and vice versa) and (2) to provide mapping and data movement on and off the Phoenix board with the lowest latency possible.

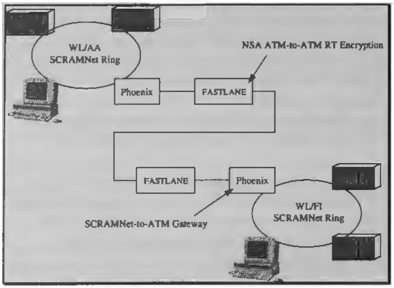


Figure 12 Using The Phoenix To Connect WL Simulations

Recall that there are three components of the overall latency between two sites. These are:

- The latency due to the movement of a word in SCRAMNet memory into the payload of an ATM cell, and the subsequent movement of that cell off the Phoenix and into the FASTLANE. And, vice versa. (x2, since there is one Phoenix at each end of the connection). Call this Latency A.

- The latency between the input and output of the FASTLANE (x2, since there is one FASTLANE at each end of the connection). Call this Latency B.

- The latency due to the path length distance between the output of one FASTLANE device and the input to the other FASTLANE device. Call this Latency C.

Figure 13 illustrates these latencies.

Our goal is to limit Latency A to between 20 and 50 microseconds, while providing a throughput of 2Kbytes at 80Hz.

It is our understanding that Latency B will be between 10 and 50 microseconds.

Latency C is the time required to move laser light through 10,000 feet of optical fiber cable. At 1.5 nanoseconds per foot, this becomes 15 microseconds. It is important to note that Latency C is installation specific.

Thus, we estimate the one-way application-to-application latency between these two specific facilities will be between 75 and 215 microseconds. (Or, 0.075 and 0.215 ms.)

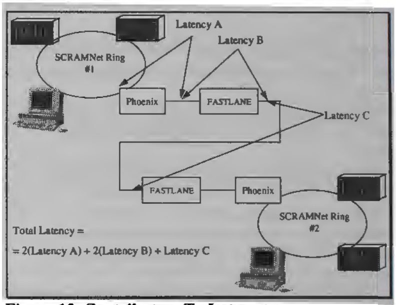


Figure 13 Contributors To Latency

The Phoenix will be demonstrated at Wright-Laboratory during 4QCY96. We plan to offer additional units beginning 1QCY97.

Summary

To summarize the major points of this paper:

(1) Unmanaged and unbounded latency will cause simulations to fail. Sometimes these failures will be obvious to all, sometimes they will be insidious and difficult to detect, and sometimes these failures will be undetected.

(2) The time-of-flight of data on the path between two distributed simulation nodes is usually not the major contributor to latency. Major contributors to latency can be attributed to bottlenecks in the software

processes and the hardware configuration used for communications.

(3) The CADIS standard of 100ms between two application layers for tightly coupled systems is unacceptable for distributed simulations of high-performance aircraft.

(4) The SNAP measurement system can accurately measure the latency between distributed simulations.

(5) "Remove the Stack" - The best way to reduce latency.

(6) "Reduce and/or Modify the Stack" - Do this if you cannot do (5).

(7) "Use Faster Pipes and/or More Pipes" - Do this if you cannot do (5).

(8) "Use Faster CPUs and/or More CPUs" - Do this in conjunction with (5).

(9) "Use the Other Machine" - Do this in conjunction with (5).

(10) "Filter, Organize, and Prioritize" - Do not generate, transmit, receive, or process information not needed.

(11) "Avoid Encryption and Decryption" - When you must use crypto devices, select a crypto system with the lowest latency possible.

Kudos

The authors thank our sponsors in the Flight Dynamics Directorate WL/FIGD), Mr. Ron Ewart, Capt. Ron Johnston, Lt. David Barnhart, Lt. Steve Purdy, Capt. David (Scott) Douglass (now in Greenland), and Mr. Barry Bryant (now at NASA Langley) for their inputs to this paper.

Author Information

George J. Valentino is a Senior Technical Manager and leader of SYSTRAN's Innovative Systems Group. He can be reached at *gvalentino@systran.com.*

Stephan A. Lubbers is a Code_Warrior (Senior Software Engineer) within the Innovative Systems Group. At SYSTRAN, he was responsible for developing the low-latency embedded software for the Phoenix SCRAMNet-to-ATM bridge. He is currently investigating the use of intelligent software agents in distributed simulation. He can be reached at *slubbers@systran.com.*

Stephen T. Thompson is another Code_Warrior within SYSTRAN's Innovative Systems Group. He architected the SNAPpy low-latency, multiprocessor network interface unit, and is currently developing multithreaded applications for SNAPpy. He can be reached at *sthompson@systran.com.*

Kenn W. Scribner is another Code_Warrior within SYSTRAN's Innovative Systems Group. His several years of flight simulator development experience is directly contributing to the SNAPpy development. He can be reached at *kscribner@systran.com.*

Kevin E. Breeding is a Senior Digital Designer within the Innovative Systems Group. At SYSTRAN, he has been responsible for the detailed engineering design of the Phoenix Single Board Computer, SCRAMNet network interfaces for PMC, GIO, and Sbus, and he is currently designing a PCI IndustryPack carrier. He can be reached at *kbreeding@systran.com.*

Bibliography

[1] 'Communication Architecture for Distributed Interactive Simulation (CADIS),' IST-CR-93-13, June 1993.

[2] 'Distributed Interactive Simulation and the Warbreaker Zen Regard Simulation, A Participant's Perspective,' Capt. Kennard W. Scribner, presented at the 1994 Royal Aeronautical Society DIS Conference, London.

[3] 'Dynamic Latency Measurement using the Simulator Network Analysis Project (SNAP),' R.Barry Bryant, Capt. D. Scott Douglass, Ron Ewart, and G.J.Slutz, Proceedings of the 16th I/ITSEC, 28 November - 1 December, 1994.

[4] 'Analysis of the Latencies in a Flight Simulator using the Simulator Network Analysis Project (SNAP),' J.M. Woodyard and Lt. D.J. Barnhart, 14th DIS Workshop, March 1996.

Note: References [2], [3], and [4] can be found at
http://www.wl.wpafb.af.mil/flight/fcd/figd/figd.html

[5] 'SCRAMNet+ Network VME-6U Hardware Reference,' Document D-T-MR-VME6U-A-0-A2, 3 August 1995.

[6] 'Reduction of PDU Filtering Time via Multiple UDP Ports,' R. Kerr and C. Dobosz, Kaman Sciences Corp., Alexandria, VA, in position paper 95-13-055 of the 13th Workshop on Standards for the Interoperability of Distributed Simulations, 18-22 September 1995

[7] Information on the Corollary C-bus II platforms can be found at http://www.corollary.com.

[8] 'Mobile Intelligence In Distributed Simulations,' Sapaty, P.S., et.al., 14th DIS Workshop, paper 158.PS.

[9] 'Communications of the ACM: Special Isue on Intelligent Agents,' Volume 37, Number 7, July 1994.

[10] Telescript information at http://www.genmagic.com

[11] Software agent archive at http://www.smli.com/research/tcl/lists/AGENTS/subject.html#505

[12] 'AI Agents in Virtual Reality Worlds: Programming Intelligent VR in C++,' Mark Watson,'' John Wiley & Sons, Inc., 1996

[13] 'Large DIS Exercises - 100 Entities Out of 100,000,' S.D.Swaine and M.A. Stapf, Proceedings of the 16th I/ITSEC, 28 November - 1 December, 1994.

[14] FASTLANE information can be obtained at http://www.gte.com/Cando/Business/Docs/Software/fastlane.html.

SCRAMNet is a registered trademark of SYSTRAN Corp.
Phoenix and SNAPpy are trademarks of SYSTRAN Corp.
SNAP is the Simulator Network Analysis Program at WL/FIGD.
C-Bus II is a trademark of Corollary, Inc.
VR-Link is a registered trademark of MAK Technology, Inc.
Windows NT is a registered trademark of Microsoft Corp.

AIAA-96-3543-CP

A REAL-TIME STOCHASTIC MTI RADAR SIMULATION FOR DIS APPLICATION

Richard Floto
Northrop Grumman Corporation
2000 W. Nasa Blvd
Melbourne, Florida 32901

Abstract

Modeling the stochastic nature of an air-to-ground radar imposes stringent demands on the processor required to implement the simulation. These demands are intensified when the simulation environment must fit the timeline of a real-time radar system. This paper examines a method to model the stochastic nature of the radar system in a Moving Target Indicator (MTI) mode using ground moving DIS targets as input. A prototype of this MTI simulation technique is being developed for incorporation of the Joint STARS radar into a Joint Advanced Distributed Simulation (JADS) environment.

One of the most important performance criteria of an MTI mode is the Probability of Detection (Pd) of ground moving targets in the presence of clutter assuming a constant false alarm rate. Pd is primarily influenced by the target returns signal-to-noise (S/N) ratio. Another important performance feature influenced by target S/N is the location accuracy of the target. This paper develops a method of characterizing target Pd and location accuracy as a function of key radar system variables that can be implemented in the real-time target detection stream of the MTI radar system.

Introduction

New advanced distributed simulation (ADS) systems are being developed with the expectation of revolutionizing the test and evaluation (T&E), and training processes. The ADS concept is to synergistically combine remotely located system and man-in-the-loop simulations so they can be exercised in concert and in real-time. The combined simulation adds affordable realism, as well as, an opportunity to evaluate developmental concepts in a realistic operational environment. The remote simulations are envisioned to be networked through the Distributive Interactive Simulation (DIS) network.

A Joint ADS (JADS) environment was developed to prove the ADS concept through the execution of an end-to-end (ETE) test scenario. The ETE replicates a complete battlefield environment, from target detection to target assignment, target engagement, and battlefield assessment. The Joint Surveillance Target and Attack Radar System (Joint STARS) E8-C platform is a principal component of the ETE test scenario due to its mission to "provide a long range airborne sensor system for standoff wide area surveillance to locate moving and stationary ground targets in support of battle management, and provide target updates for effective and efficient target attacks." The Joint STARS radar will integrate live targets with simulated virtual targets from the DIS in a seamless manner.

Figure 1 shows a typical JADS environment for the ETE test scenario. The E8-C will fly over a test facility where a limited number of controlled targets will be located. A remotely located target/war simulation as JANUS will provide virtual targets onto the DIS through the use of Protocol Data Units (PDU). The Joint STARS radar will enhance the virtual targets for realism by introducing MTI Pd (probability of detection), CEP (circular error probable), false alarm and terrain screening effects. The enhanced target reports are combined with the appropriate live targets for output from the E8-C. This scenario requires the Joint STARS radar system to handle at least 5000 virtual targets in addition to the normally detected live targets under the real-time constraints of the Joint STARS system.

Probability of Detection

This section will define in four parts how target Pd statistics are applied to ground moving DIS targets. The first part of this section develops the relationship of the DIS target to the Joint STARS radar. The second part develops the fundamental relationship of S/N to Pd for the Joint STARS radar. The third part shows how the first two parts may be used to develop a real-time MTI Pd simulation. The last part shows typical results of the real-time MTI Pd simulation.

Geometrical Filtering of DIS targets

Prior to applying Pd statistics to the moving targets, the geometrical relationship of the target to the radar antenna must be determined. The set of ground moving targets received from the DIS network must first be dead reckoned to the current dwell time so that they accurately represent the target's true position. These targets are then filtered to determine which targets reside in the current radar beam footprint. This process can be done by either converting the target from its earth-fixed coordinate system to the radar's polar coordinates or by converting the radar beam information to the target's earth-fixed coordinate system. Since the simulation must handle a very large set of targets and the targets must eventually be reported by the radar in earth-fixed coordinates, the latter approach was chosen in the simulation to save CPU time.

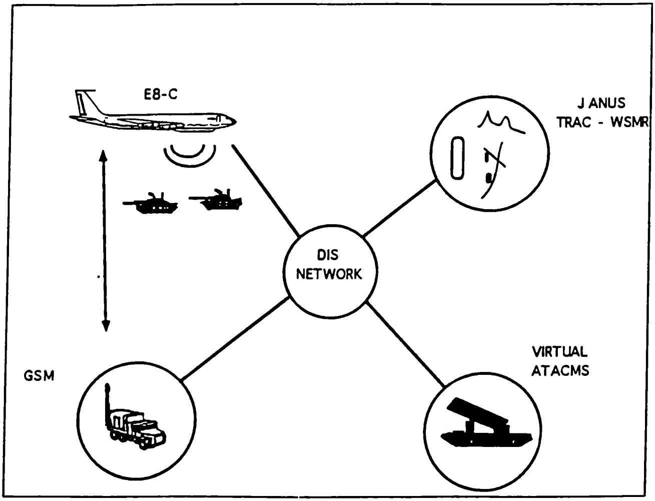


Figure 1 - Typical JADS Environment

S/N Effects on Target Pd

As previously stated, target Pd can be fundamentally characterized as a function of S/N, and is traditionally developed using the basic radar range equation[1] and subsequently modified to include radar system specifics. For air-to-ground radars, the presence of ground clutter adds a significant complication to the process of detecting a target and must be addressed when developing a design to meet a required Pd. For the Joint STARS system, the probability of detecting a moving target must be considered as a function of target-to-noise (T/N) and target-to-clutter (T/C), where the sum of noise and clutter is considered the interference in the target detection process. Since the detection of moving targets relies upon the Doppler effect, Pd can be developed as a function of T/N and T/C of the Doppler filter in which the target resides.

A set of analytical tools have been developed to measure and predict target Pd for the Joint STARS system under the conditions in which the system is required to operate. One such tool simulates the transmission, reception and processing of a number of coherent processing intervals (CPIs) comprising a set of different pulse repetition frequencies (PRFs) of specified integration length and azimuth beam spacing used to detect and locate a moving target in a clutter background. The target is placed in the desired range and angle location and evaluated over a range of radial velocities. For each CPI, the T/N and T/C ratios are determined for each Doppler filter by convolving the Doppler filter spectrum against the combined clutter, noise and target spectra. From the combined clutter and noise environment, a detection threshold is determined that will allow the system probability of false alarm (PFA) to be met. A Swerling I[4] target model is used with the detection threshold to determine target Pd for each CPI. Assuming each CPI represents an independent look at the target, a composite target Pd is established over 'N' CPIs using an 'M' out of 'N' detection scheme. These analytical tools are non-real-time in nature and have been used in the development of this real-time simulation of Joint STARS MTI.

Real-Time MTI Pd Simulation

Most of the terms in the radar range equation have a fixed allocation and are used primarily as a reference point for Pd determination. However, there are two important loss factors which must be accounted for due to the physical nature of the system; antenna beam broadening loss[2]; and target range. The antenna beam broadening loss results from electronically steering the antenna beam in azimuth and is shown in figure 2. Target range represents the largest S/N impediment to target detection at long ranges due to the two way nature of radar resulting in a 40Log(R_t) loss term. These two features must be compensated for in real time by the Joint STARS radar in order to meet the system's specified Pd. This is accomplished by increasing the radar's time on target in both the range and azimuth direction through an optimal combination of beam spacing and integration time. This combination can be normalized into a single term known as the 3dB dwell time (i.e., the time spent on a target between the half power points of the radar beam).

In order to develop a simulation which inputs ground moving DIS targets into the real-time data stream, the target Pd model must be computationally efficient as well as representative of the conditions from which the scan was initiated. Using the analytical tools described in the previous section, a database of radar operating curves have been developed which define the relationship of average Pd as a function of azimuth, range and dwell time. Since the curves are reasonably well behaved, they have been defined as a Lagrange interpolation series of four points[3]. Figure 3 shows how the 3dB dwell time varies as a function of radar scan angle to meet a specified average Pd, compensating for the azimuth beam broadening loss shown in figure 2. Figure 4 shows how the 3dB dwell time for a given scan angle varies as a function of range to meet a specified Pd, compensating for the two way range loss. The resultant 3dB dwell time may be compared against the radar's actual 3dB dwell time and interpolated between the family of Pd curves to obtain the average Pd of the DIS ground mover given its geometrical relationship (i.e., range and angle) to the radar.

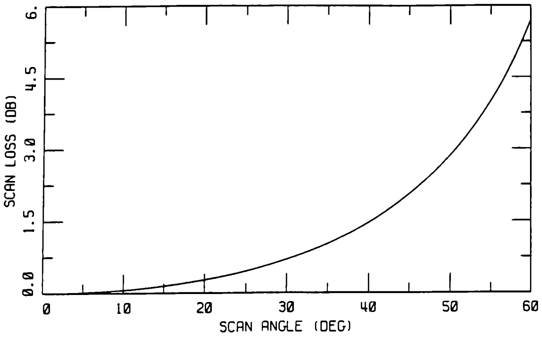


Figure 2 - Azimuth Beam Broadening Loss

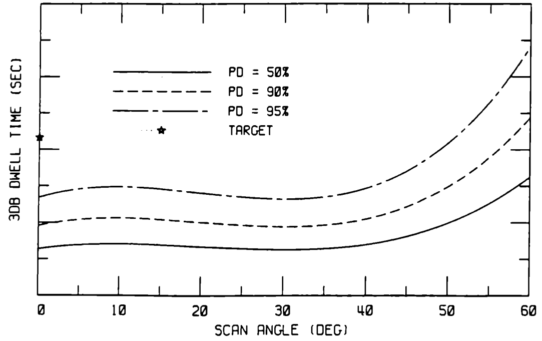


Figure 3 - Expand Dwell Time by Azimuth
(@ System Minimum Range)

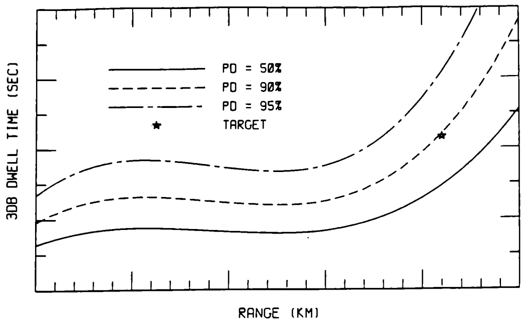


Figure 4 - Expand Dwell Time by Range
(Scan Angle = 0 degrees)

The final part of the MTI Pd simulation is the application of target radial velocity to the Pd process. Low PRF (range unambiguous) MTI radars having ambiguous Doppler measurements, referred to as blind speeds, are caused by the Doppler frequency shift near multiples of the PRF[1]. The blind speeds (V_n) for a PRF are:

$$V_n = \frac{1}{2} \lambda \, n \, PRF \qquad n = 1,2,3...$$

These blind speeds are reduced by employing a multiple PRF design like that used by Joint STARS. A multiple psuedo-low PRF design used by Joint STARS has been selected as a tradeoff between range and Doppler ambiguities, so as to optimize the ability of the radar to detect and disambiguate the velocity ambiguities to resolve the radial velocity of the target.

Figure 5 shows the Pd performance of two different combinations of multiple PRF designs operating at the same range, angle and S/N conditions. Note that the structure of the curves over velocity are completely different, but the mean Pd with respect to velocity is approximately the same. Figure 6 represents the Pd performance of the same combinations of PRFs, but with different S/N ratios. Note that the structure of the Pd curves in figure 6 are identical and that the Pd dips with respect to velocity are exaggerated at lover Pd levels. Since the structure of these curves are difficult to accurately model in real-time, the MTI Pd simulation will store a reference curve for each combination of PRFs used by the Joint STARS radar. Other Pd curves can be derived from the reference Pd curve as follows:

$$Pd_t = \overline{Pd}_t + \dot{R}_t \, (Pd_n - \overline{Pd}_r)$$

where,

Pd_t = Pd of target at velocity V.

$\overline{Pd}_t$ = Pd of target averaged over all V.

$\dot{R}_t$ = Range Rate Factor to enhance Pd dips.

Pd_n = Pd of reference curve at target V.

$\overline{Pd}_r$= Pd of reference curve averaged over V

Real-Time MTI Pd Results

The example given in figures 3, 4, and 7 show a target located broadside (0°) and at a specified range (R_t) and velocity (V_t). Figure 3 states that if the target were at the system's minimum range, the target's expected Pd would easily exceed 95% and in fact approach unity. Note that the expected Pd would be 95% if the target was located at 50° azimuth and 90% at 60°. Figure 4 develops the range versus dwell time curve based on the target's azimuth position of 0°, so that in this example the target's average Pd (wrt velocity) is 90% at range R_t. Figure 7 shows how the target velocity Pd dips are applied from the reference curve to predict the target's actual Pd at the specified velocity. In this example, the target's Pd is predicted by the real-time simulation to be within 2% of the reference simulation for the identical range, angle, velocity and S/N conditions.

In actual practice, the real-time Pd simulation must declare a target detected or not-detected. This is accomplished by comparing the derived target Pd against a uniform random number sequence ranging from 0 to 1. If the random number exceeds the derived Pd, then the target is declared a miss, otherwise it is considered a hit. Figure 8 shows the results of the real-time simulation applied over several hundred scans of a target positioned at various ranges. These results are superimposed onto the reference simulation results to show a high correlation between the real-time simulation and the expected system performance. The results of this method demonstrate that an accurate simulation of radar MTI performance can be modeled in real-time given the radar dwell time and the target's range, azimuth and velocity.

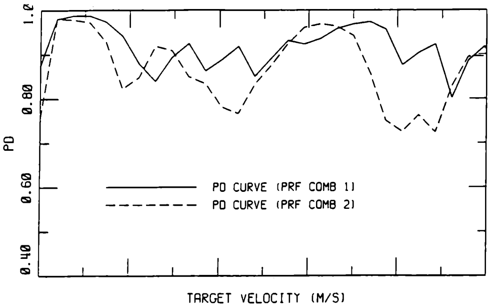


Figure 5 - Constant S/N for Different PRF Combinations

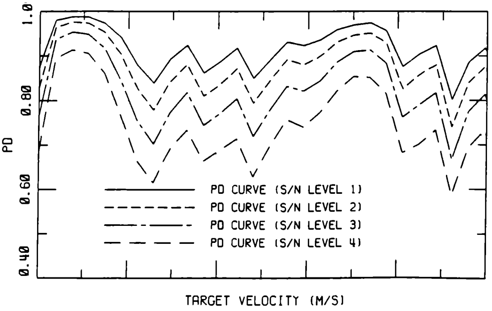


Figure 6 - Vary S/N for Same PRF Combinations

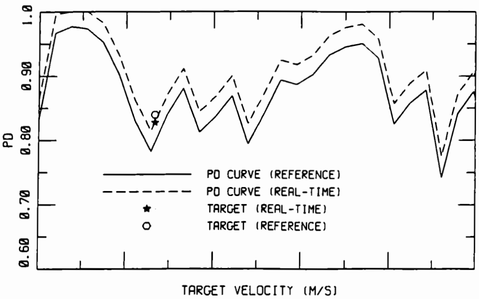


Figure 7 - Apply Target Velocity Pd dips

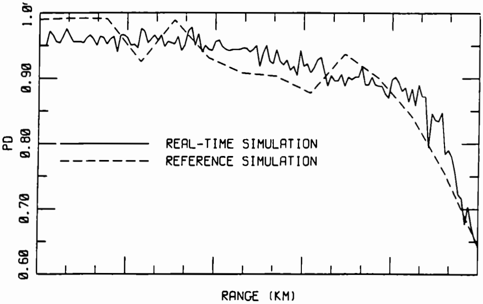


Figure 8 - Real-Time vs Reference Simulation

Location Accuracy

The previous section developed how a DIS target is detected by the Joint STARS radar. This section will define in three parts how the appropriate location accuracy statistics are applied to the detected DIS targets to make them appear to be Joint STARS targets. The first part defines the measurement characteristics of target location using the circular error probable (CEP) method. The second part defines some of the error sources affecting the location of the target based on the target's S/N. The last part shows the typical results of the real-time MTI CEP simulation.

Circular Error Probable (CEP)

The CEP integral has been developed for a variety of systems ranging from assessing when to fire artillery by projecting targets into the shell's "kill-zone", to the evaluation of radar systems and wind sheer[5]. CEP is defined as the radius for which 50% of the targets fall within the radius, and 50% fall outside this radius. In the case of evaluating the location accuracy of a target detected by a radar, the CEP can be derived from the relationship of the 1-sigma down range (S_{dr}), and 1-sigma cross range (S_{cr}) errors as shown in figure 9[6].

S/N Effects on Target Location

There are four classes of error sources which contribute to the location error budget for MTI targets. The first class of errors result from an incorrect range measurement and can be attributed to range resolution, atmospheric refraction and hardware errors. The second class of errors results from incorrect angle measurements due to thermal noise, false patch clutter, hardware and line-of-sight velocity errors. The third class of errors result from vertical separation uncertainties due to platform altitude and ground terrain errors. The last class of errors result from a coordinate system error, resulting from navigation and coordinate conversion errors. These error sources form a linear error model having two outputs (a down range and a cross range error) from which the target's CEP may be estimated[6].

Target S/N has an affect on the angle accuracy of the target. Thermal noise induces an error in the interferometric measurement which is inversely proportional to the square root of the T/N ratio. False patch clutter influences angle accuracy because moving targets compete against clutter from a different azimuth angle than the target. This error is inversely proportional to the square root of the T/C ratio. As with target Pd, curves can be developed which relate angle error as a function of target S/N resulting from a specified dwell time, range and azimuth.

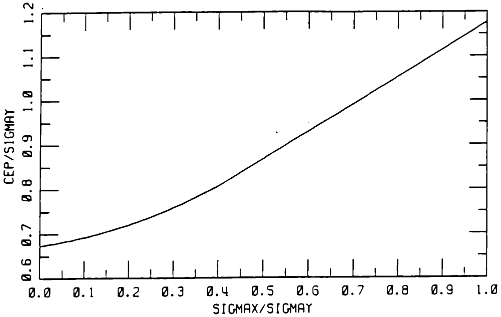


Figure 9 - Circular Error Probable (CEP)

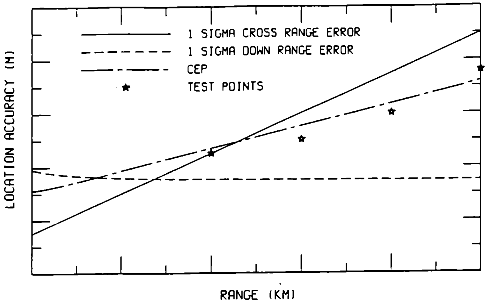


Figure 10 - CEP Approximation vs Range

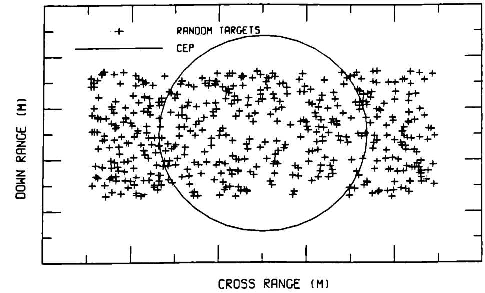


Figure 11 - Target 4 CEP

Real-Time MTI CEP Results

The example given in figure 10 shows the expected CEP as a function of range, as derived from the 1-sigma down range and cross range errors. Superimposed on the CEP curve are four targets placed at different ranges such that the actual 1 sigma down range and cross range errors have been randomly selected over several hundred scans of the target. These results indicate a high correlation between the real-time MTI CEP simulation and the derived CEP values. Figure 11 shows the random selection of down range and cross range errors for one of these targets.

One drawback to selecting location errors randomly is that the target appears to have a random fluctuation within a predetermined boundary and can make the target appear too "simulation like". One method used to make the target appear more realistic, is to use a relatively narrow error boundary whose mean is modulated in both amplitude and frequency. This technique has the effect of moving a smaller CEP circle within the larger true CEP circle to achieve the same CEP results with fewer unrealistic random fluctuations.

Conclusion

In this paper, a technique for realistically modeling ground moving DIS targets in a real-time MTI radar system has been presented. Both target Pd and CEP can be accurately modeled in a large scale simulation for a real-time system. Timing studies conducted on radar beam dwells with densely packed targets indicate that the MTI simulation requires only 25% of the dwell time to apply the MTI statistics to the targets, so that the remaining timeline can be used for other simulation effects. The results of this study indicate that large scale ADS systems can be realistically implemented by host systems in a real-time environment.

References

[1] M. I. Skolnick, Introduction To Radar Systems, second edition. New York: McGraw-Hill Book Company, 1980, pp. 23-32, 108, 114-117.

[2] G. W. Stimson, Introduction To Airborne Radar. El Segundo: Hughes Aircraft Company, 1983, pp. 139-141.

[3] I. A. Dodes, Numerical Analysis for Computer Science. New York: Elsevier North-Holland, Inc., 1978, pp. 184-194.

[4] P. Swerling, Probability of Detection for Fluctuating Targets. IRE Trans., vol. IT-6, April 1960, pp. 269-308.

[5] J. T. Gillis, Computation of the Circular Error Probability Integral. IEEE Transactions on Aerospace and Electronic Systems, vol. 27, no. 6, November 1991, pp. 906-910.

[6] J. S. Przemieniecki, Introduction to Mathematical Methods in Defense Analyses. Washington DC: American Institute of Aeronautics and Astronautics, Inc., 1990, pp. 35-49.

AIAA-96-3544-CP

Low cost, high fidelity Flight simulation environment

Richard Tremblay
Flight Simulation
Virtual Prototypes Inc.
Montreal, Canada

ABSTRACT

As the need to reduce costs in every aspect of flight simulation becomes of utmost importance, the use of commercial-off-the-shelf (COTS) software components to improve both the design and the implementation phases of the development of flight trainers or avionics test benches is necessary. This paper presents the FLSIM COTS package which can be readily transformed into a high fidelity flight simulation environment with minimal investments in terms of software development effort. The package is highly flexible and can be adapted to all aircraft simulation by non-programmer flight dynamics personnel.

1. INTRODUCTION

The increased complexity of military weapons systems combined with the requirements for life cycle upgrades and reduced initial development costs has generated the need for new integration concepts. The life cycle of these systems is overwhelming because of the unmaintainability and the prohibitive cost of upgrades. The solution is to develop an integration concept which allows evaluation of new technology products without complete system reconfiguration. In the avionics upgrade programs, COTS (commercial off-the self) software packages address two important common problems. They reduce the cost of initial requirement specifications and provide economical engineering tools for the design and the life cycle upgrades.

In both military and civilian aircrew training, it has been common to develop high cost, custom designed components in order to meet the training objectives and requirements. There have been considerable efforts put into the development of standards aimed at reducing these high cost. Others have put efforts in the modularization and the reuse of software source code. However, the trends over the past few years has been to move towards the inclusion of COTS components within training systems in order to reduce cost and life cycle maintenance as well as providing easy system upgrade paths.

The ultimate objective of any modern training system is to provide the most effective training for the minimum investment. Consequently, COTS components which would fulfil the following key features would provide the answers to these high cost problems:

- Ease of operation
- Growth potential both in terms of performance and flexibility
- Highly flexible reconfiguration
- Maximum use of off-the-shelf hardware workstations
- System operation with minimum hardware expertise
- Ease of customization by subject matter experts (SMEs)
- Natural integration with third party software/hardware components (visual, sound, MIL-STD-1553)
- Openness
- Selective replacement of core simulation models with real avionics equipment

2. THE LOW COST SOLUTION

Traditionally, developing the software system of any flight simulator has been one of the highest cost elements. Engineering hours normally outweigh manufacturing and production hours. The description of kinematical and dynamical equations of flight specific to different types of aircraft is an integral part of the development process of flight simulators and avionics test rigs. Although the equations of motion have been known for a number of years, customizing the forces and moments acting on one specific aircraft can become cumbersome and prolonged. Furthermore, evolutionary rapid prototyping is now becoming a standard in several organizations dealing with avionics development. Using reconfigurable cockpits, avionics developers, system integrators and training organizations, are providing cost effective solutions to the total system development life cycle. For these organizations, a COTS package which offers a user friendly interface for non-programmer users, reconfigurability and flexibility for R&D environments, and reliable real-time performance for training applications is certainly of great interest.

Normally, a mathematical model is developed for the physical system (the aircraft response) using knowledge of the physical laws describing the system (aerodynamic coefficients). This model is then programmed on a computer in order to resolve the problem. With the FLSIM commercial-off-the-shelf package, once the design of the aerodynamic model has been completed, there is no need for coding. FLSIM will use the sequence of coefficients that was designed to resolve the problem.

The FLSIM COTS package was designed to expedite the development of low cost, high fidelity flight simulation programs. Even though generic in nature, it transforms the aerodynamic coefficient curves provided by the user into realistic flight simulation which matches the performance of the desired aircraft. FLSIM will let the end user concentrate on tailoring the behaviour of the aircraft by entering flight test or design data for the different aerodynamic coefficients instead of spending its time on writing source code for the solution to the different aerodynamic equations. The primary objective of the package is to provide a user friendly, flexible, and reliable real-time environment for high fidelity flight simulation. It provides an environment for configuring any fixed-wing aircraft using definable flight parameters and allows the resulting flight model to be flown across any image generator.

Its basic open and modular architecture will support future growth through the addition of new modules or functions. The core flight model program can also be used in a number of different applications ranging from avionics integration test benches to Part Task Trainers (PTTs) through cockpit layout and Multi Function Displays (MFDs) design human interaction studies. All these applications can be accomplished using the same commercial off-the-shelf hardware which is another key feature of FLSIM. Its internal structure makes it suitable to be used as a standalone software product or as part of a complete turn key solution for R&D or training environment.

FLSIM also comes with a development library should the end user wish to append functionality not provided by the base program, or modify its behaviour and therefore achieve a higher level of customization.

The approach taken by FLSIM for simulating aircrafts is very simple. It involves three (3) types of users:

- A 'Modeller' uses FLSIM to enter the aerodynamics parameters for a particular aircraft and achieve a very high degree of aerodynamic fidelity. This is the heart and it is where the generic COTS package is transformed into a high fidelity specific flight dynamics model. This aspect of the product will be detailed further below in the document.

- A 'Developer' can add functionality to the base COTS package or replace some of the core models to achieve an extremely high level of fidelity in some systems simulation. This is also where a very close coupling with other major subsystems would be established.

  A few examples of the utility of a development environment are:

  - replacing the "flight control computer simulation module" in order to replicate the exact behaviour on board a specific aircraft

  - adding the simulation for a fire control radar

  - providing an interface to custom hardware like joystick, throttle, sound generators, control loading system

- The 'End User' uses the final product to flight different missions in a flight trainer or performs diverse tests in an avionics test bench.

As can be expected, each task is suited to different interests and skills. This approach lets each 'domain expert' work in its discipline. Each category requires the competence appropriate for the task. It allows the modeller to concentrate on aerodynamic and not have to become a programmer. It allows programmers to focus on writing the code for a particular aircraft subsystem without becoming an authority in aerodynamic. Finally it permits the end user to direct their attention to flying the resulting aircraft.

3. OVERALL ARCHITECTURE

The FLSIM package is divided into 3 distinct processes:

- the user interface, where the flight model is defined and the data is entered and saved in the database
- the simulation models
- the Out-The-Window (OTW) scene

The communications between the three processes is accomplished via Ethernet with the TCP and UDP protocols of the IP family. This scheme enables the user to locate the simulation models in any workstation in the system, where the best coupling with the rest of the simulation complex can be achieved. It also enables the user to replace the standard FLSIM user interface by a custom interface which would be specifically designed for the application. One could use the standard Man Machine Interface (MMI) to build the intended aircraft dynamics and then use the custom MMI to control the target system. In either case, the simulation models will not distinguish between the two.

When FLSIM goes into run-time mode, the scene process is started. It shows a simple runway with a Head-up Display (HUD) overlay. It is basically used for VFR (visual flight rules) flying. It receives the run-time data from the simulation and displays the pilot point of view accordingly. Because the scene is a separate process, it can easily be replace by a third vendor scene generator. One has only to change the name of the OTW process in one menu of the user interface.

4. THE SIMULATION PROCESS

The simulation process is basically subdivided into the following 3 major segments:

- the run-time controller (RTC)
- the simulation models
- the run-time data exporter

The run-time controller

The RTC is responsible for ensuring that all simulation models are executed in a predefined sequence. The RTC supports the following execution modes:

Synchronous Mode

In this mode, the RTC ensures the sequential execution within fixed time intervals (real-time simulation). The constant sequencing of the models (time interrupts) is maintained by the workstation hardware clock (Figure 1).

Asynchronous Mode

In this mode, the RTC controls the sequential execution in a "free-run" mode, i.e. without time constraints on the execution cycle. It simply waits user specified number of milliseconds between the end of one iteration and the start of the next one. This mode becomes very significant and valuable when developing new application modules (Figure 2).

Slave Mode

This is the same basic idea as the synchronous mode except that the time interrupts will be provided by another process on the workstation. This mode is very useful for synchronizing several process (Figure 3).

User Dispatch (linked) Mode

In this mode, the simulation models are embedded within a user developed main process. Therefore the dispatching of the simulation models is user dependent. This enables the FLSIM simulation models to be included within a user program either written in C/C++ or even in Ada (Figure 4).

In all of the modes, the **Simulation Control Process** (see figures 1 to 4) can either be the normal FLSIM user interface, or a user program which is built from the FLSIM development libraries.

The simulation models

The FLSIM simulation models logically divided as follows:

- earth
- atmospheric
- air data computer
- pilot inputs
- autopilot logic
- autopilot flight management system
- autopilot flight control computer
- autopilot inner stability loops
- control surfaces
- engine (turbojet, turboprop, piston)
- fuel system

- additional loads
- aerodynamics
- undercarriage (gear, brakes)
- weight and balance
- equations of motion, rotational
- equations of motion, translational
- attitude
- position
- NAVAIDS
  - VOR
  - DME
  - TACAN
  - NDB
  - ILS
- flight instruments
- DIS
- malfunctions

The objective of this grouping is to provide an easy understanding of the models and to provide the capability of replacing any combination of the simulation models by user supplied models.

The run-time data exporter

FLSIM can output the run-time data to external processes and shared memory. External processes can register to receive the UDP/IP data packets. The data packets contained all the information necessary to drive a basics scene, sound generator and cockpit instruments.

5. THE USER INTERFACE PROCESS

The user interface process provides the interaction with the FLSIM database. FLSIM provides an advanced user interface to define the parameters of the flight model, engine model, atmospheric model, initial conditions and waypoint trajectories for the flight management simulation.

The FLSIM User Interface incorporates the following features:

- rationalized user interface to group related functions and provide an overall well organized control scheme
- extensive data definition interface
- data definition is supported through numerical data entry or graphical curve definition using the mouse as a pointing device
- file selection menus to facilitate the loading of existing data files stored in the database

In addition, all the data can be viewed with any units the user feels comfortable with. The units in the User Interface can be customized by simply editing an ASCII file.

6. MODELLING THE FLIGHT DYNAMICS

The heart of FLSIM is its reconfigurable flight dynamics model. The modeller communicates with FLSIM by providing the aircraft physical and aerodynamic parameters. That person must have a fairly good understanding of aircraft behaviour and aerodynamic terminology. However, it does not have to be an expert in flight dynamics. Although an understanding of the original control and stability derivative data package is all that is really required, to be able to use FLSIM to its maximum, a good knowledge of the interaction between the aerodynamic coefficients and the aircraft response is a desired asset.

Flight data packages come in different flavours and come from a number of sources (flight test, wind tunnel, design data, DATCOM techniques or virtual wind tunnel computer programs). Each aircraft manufacturer or agency contracted to assemble a data package presents the results in unique ways. In addition, turbofan or turboprop driven aircraft, large transport or small jet fighters will most likely be represented by different data packages. Lets look at the untrimmed lift coefficient from three sources which shows the problem that the COTS product is faced with:

B747 data package [1]

$$\begin{aligned} C_{L_{untrimmed}} &= CL(\alpha_{WDP}, \delta_f, Mn) \\ &+ \Delta CL_{\alpha_{WDP}=0}(Mn, \delta_f, h) \\ &+ \Delta \frac{\partial CL}{\partial \alpha}(Mn, \delta_f, h) \cdot \alpha_{WDP} \end{aligned}$$

C130 data package [2]

$$\begin{aligned} C_{L_{untrimmed}} &= CL(\alpha_{FRL}, TC, \delta_f) \\ &+ \Delta CL(\alpha_{FRL}, Mn) \end{aligned}$$

C141 data package [3]

$$\begin{aligned} C_{L_{untrimmed}} &= CL_0(Mn,\bar{q},\delta_f) \\ &+ CL_\alpha(Mn,\bar{q},\delta_f)\cdot\alpha_{FRL} \\ &+ CL_{stall}(\alpha_{FRL}) \end{aligned}$$

where:

α_{FRL}	angle of attack, fuselage reference line
α_{WDP}	angle of attack, wing design plane
TC	turboprop thrust coefficient
Mn	mach number
δ_f	flap deflection
$\bar{q}$	dynamic pressure
h	barometric altitude

It becomes evident from these examples that something simple as the basic untrimmed lift coefficient is expressed in totally different formats. Furthermore, the same variable can be represented with different reference as is the case for angle of attack. Some data packages have even used both types of angle of attack reference. It would be very difficult for a COTS package to have built-in curves that would meet the requirements presented by everybody's data packages. This is the way that FLSIM versions 4.0 to 4.2 were originally designed. It contained a predefined set of curves and the build-up of each coefficient was fixed. It became rapidly evident, even though the results were satisfactory, that such restrictions would prohibit its inclusion within full flight simulators. This is the reason why version 4.3 of the FLSIM flight model was totally revised. Instead of having the end user try to match its original data package to the FLSIM curves, therefore losing and maybe introducing errors and loss of precision in the conversion process, we provided the capability of building the flight model to match the data package.

The definition of a specific flight model consists of three simple steps:

- obtaining the control and stability derivative data for the desired aircraft
- from the FLSIM user interface, creating the list of terms (by simple point and click interaction) that will define the aerodynamic coefficients curves corresponding to the manufacturer data for each of the 6 coefficients (Lift coefficient C_L, Drag coefficient C_D, Side Force coefficient C_Y, Rolling Moment coefficient C_l, Pitching Moment coefficient C_m, Yawing Moment coefficient C_n)
- from the FLSIM user interface, inputting the data graphically by simple point and click actions within the pulldown menus and popup windows using the mouse and the keyboard. Data entries consist on input fields, selection menus and graphical curves.

Description of Coefficients

Each one of the 6 aerodynamic coefficients will be built of a collection of terms summed together:

$$C_x = term_1 + term_2 + \ldots.. + term_n$$

where $term_x$ is a block of 4 items which can be +, -, *, / together.

Each of the item can be any of the following:

1. constant
2. parameter
3. curve
4. sub-block (4 items with +, -, *, /), the items can be any of the 5 types listed
5. reference to another term in any of the 6 aerodynamic coefficient term sequence

The curves provide the capability for 1, 2, or 3 independent variables.

$$x = F(parameter_1)$$
$$x = F(parameter_1\ ,\ parameter_2)$$
$$x = F(parameter_1\ ,\ parameter_2\ ,\ parameter_3)$$

For these curves, the user has the capability of specifying the minimum value, maximum value for the curve axes so the displayed range is appropriate to the data being presented.

The parameters (used in curves and coefficient terms) can be of 2 types. FLSIM provides an exhaustive list of standard parameters. In addition, user parameters can be built from these standard parameters by adding, subtracting, multiplying, and dividing them together. The standard parameters are:

α_{FRL}	angle of attack, fuselage reference line
α_{WDP}	angle of attack, wing design plane
$\dot{\alpha}$	rate of change of angle of attack

β	angle of sideslip
$\dot{\beta}$	rate of change of angle of sideslip
δ_{a_l}	left aileron deflection
δ_{a_r}	right aileron deflection
$\delta_{a\_trim_l}$	left aileron trim deflection
$\delta_{a\_trim_r}$	right aileron trim deflection
δ_r	rudder deflection
δ_{r_trim}	rudder trim deflection
δ_e	elevator deflection
δ_{e_trim}	elevator trim deflection
δ_{hs}	horizontal stabilizer deflection
δ_{f_l}	left flap deflection
δ_{f_r}	right flap deflection
δ_{n_l}	left leading edge flap deflection
δ_{n_r}	right leading edge flap deflection
δ_{sp_l}	left spoiler deflection
δ_{sp_r}	right spoiler deflection
δ_{g_n}	nose gear position
δ_{g_m}	main gear position
δ_{sb}	speed brake deflection
p_s	stability axis roll rate
q_s	stability axis pitch rate
r_s	stability axis yaw rate
p_b	body axis roll rate
q_b	body axis pitch rate
r_b	body axis yaw rate
h_{rad}	radar altitude
h	barometric altitude
$\dot{h}$	vertical velocity
$\bar{q}$	dynamic pressure
Mn	mach number
V_t	true airspeed
V_i	indicated airspeed
V_e	equivalent airspeed
b	wing span

$\bar{c}$	mean aerodynamic chord
TC	thrust coefficient
FT	jet thrust force
m	aircraft mass
N_z	normal force
C_L	Total lift coefficient
C_D	Total drag coefficient
C_Y	Total side force coefficient
$\Delta_{cog_{FS}}$	Distance between aerodynamic reference position and center of gravity along the Fuselage Station axis
$\Delta_{cog_{BL}}$	Distance between aerodynamic reference position and center of gravity along the Butt Line axis
$\Delta_{cog_{WL}}$	Distance between aerodynamic reference position and center of gravity along the Water Line axis

It is believed that this comprehensive list of parameters together with the definition of curves and coefficient terms will enable the simulation of most flight data packages by non-programmer subject matter experts. This was one of the features required by the desired COTS product.

A pictorial representation of a coefficient term is presented in Figure 5.

7. DEPLOYED SYSTEMS

The following list provides some of the projects that FLSIM was successfully used on:

- Republic of Korea Air Force Academy, pilot aptitude research equipment
- EASAMS in Australia, F18 support facility
- Northrop B-2 division, Electronic FLIP (flight information publications) displays demonstration
- Alta company, Level C flight simulator (in development)
- Lockheed/Martin Aircraft Services
  - A4 upgrade
  - C130 demonstration

8. CONCLUSION

This paper has demonstrated that COTS packages can be very effective in today's flight simulation environment. The FLSIM COTS product previously described can provide powerful an innovative training systems and avionics integration test rigs which offer a the following features through the use of advanced technologies:

- reduce costs
- flexibility
- ease of customization
- natural integration with other software/hardware
- growth potential

The FLSIM low cost, high fidelity flight simulation environment can only be beneficial to the flight simulation industry.

REFERENCES

[1] The simulation of jumbo jet transport aircraft, Volume 2.: Modelling data, Boeing company, 1970

[2] Report no LG81ER0117, Aerodynamic coefficients C-130H, Lockheed-Georgia company.

[3] Flight test report of the USAF C-141B ATS upgrade, Volume 1, Kohlman Systems Research Inc., 1990

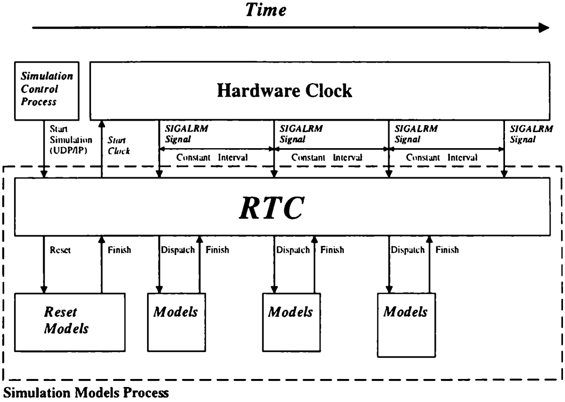


Figure 1 Synchronous Mode

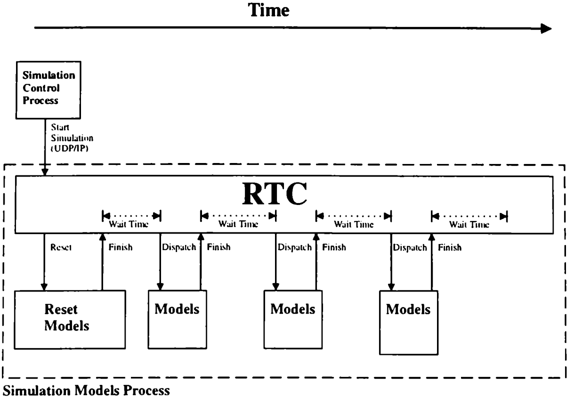


Figure 2 Asynchronous Mode

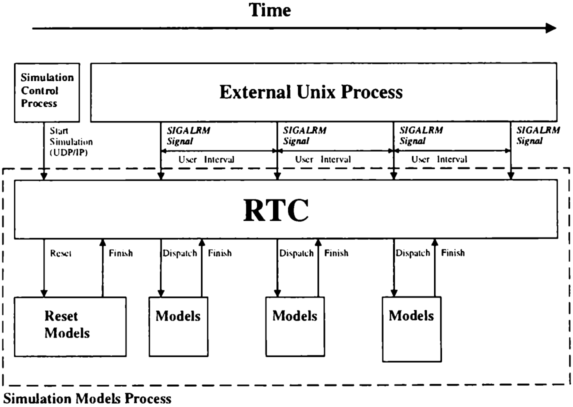


Figure 3 Slaved Mode

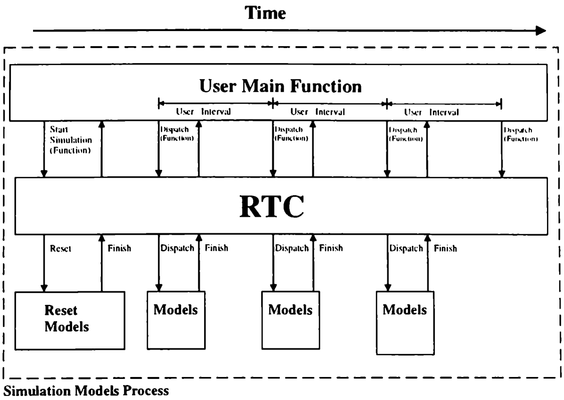


Figure 4 User Dispatched Mode

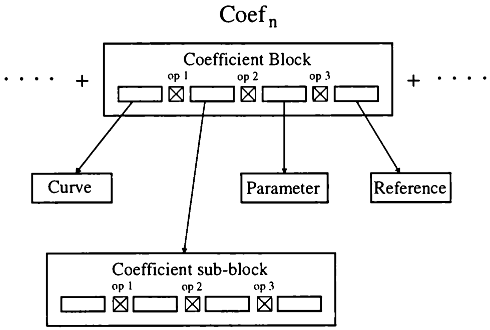


Figure 5 Aerodynamic Term

AIAA-96-3545-CP

CREATION OF VARIABLE GENERIC DATABASES IN REAL-TIME

S Biggs
Product Manager Image Generation
Thomson Training & Simulation Ltd.
Gatwick Road, Crawley, W.Sx.
United Kingdom

Abstract

One of the most important features of a modern flight simulator is the ability to create visual training databases quickly and at low cost. Tools which provide for rapid creation of generic databases, representative of different natural environments around the world enable civil airlines to accumulate a large library of secondary diversion airfields that can be used in line oriented flight training (LOFT) exercises. It also produces the possibility of very rapid mission rehearsal preparation in military applications. To facilitate these tools, a fully depth (Z) buffered visual architecture is required. This allows an object based Variable Generic Database facility to have all occulting considerations handled by the image generator in real-time allowing complete freedom of positioning and orientation of the components of the database without the constraint of potential occulting errors. Variations in training conditions can be simulated on demand from the Instructor Operating Station (IOS) on the simulator in real-time through the provision of powerful user friendly interactive database design tools. The result is that training quality generic databases can be built and compiled rapidly, ready for immediate use on the simulator.

Introduction

Simulation and training operators are continuously searching for ways of reducing the levels of investment required to build up a library of airfields for comprehensive training. Creating visual training databases is traditionally a high cost, time consuming activity, requiring a degree of expertise and modelling training in the use of the latest modelling tools available.

Generating fully-customised models of all airfields is not cost-effective, especially considering the fact that while mainstream training will be restricted to a limited number of airports, emergency conditions could dictate diversion to any airport, anywhere in the world.

Generally, it is not necessary to create highly detailed models of these secondary diversion airports, but simply to provide representative terrain of the surrounding environment and enough buildings to identify the layout of the airfield. It is however, important to provide accurate runway layout and lighting systems and key landmarks on the final approach and glide slope to the runway.

Based on these requirements and constraints, Thomson Training and Simulation have developed a set of tools which provide for cost effective creation of generic databases that can be modified in real-time for immediate training use. Described here are an Automatic and a Manual Variable Generic Database utility, which allow the creation of generic airfield models, either off-line or in real-time, from a set of simple building blocks.

Manual or Automatic?

So as to provide optimum training efficiency, two types of Variable Generic Database have been developed;

- Automatic Variable Generic Database (AVGDB) for creating databases on demand from the simulator, which are specified by a subset of the simulator ground station data.

- Manual Variable Generic Database (MVGDB) for rapid database creation either off-line or on-line using a library of pre-modelled objects.

AVGDB

Built on an environment model with a flat terrain in the airfield vicinity, the Automatic Variable Generic Database (AVGDB) provides a choice of environments. These include some terrain features where appropriate to the generic nature (such as low rolling hills, sand dunes, jungle, urban sprawl, etc.). The airfield is built on a platform that may be tilted to give the whole airfield a slope. Runway width, length, markings, approach light patterns, taxi-way arrangement, etc. are selectable from a library of objects pre-modelled.

The environment and airfield parameters for both day and dusk/night scenarios are specified by a well defined set of options which can be pre-programmed in up to 6000 or more possible runway definitions world-wide (one for each runway end). The number of AVGDB parameters actually available is limited only by simulator storage considerations.

An AVGDB is loaded in the same way as a custom database, during training, by selecting the terrain object from the IOS. Selection of the environment type and airfield parameters can then rapidly be made by activating animations from the IOS. Table 1 shows a breakdown of the main features controllable.

Table 1: AVGDB Controllable Features

Parameter	Specification
Environment terrain	8 generic types; green rolling hills, desert, jungle, etc.
Environment lighting	2 types (US, UK)
Runway approach lighting system (ALS)	14 types (including TDZ)
Terminal and taxi-way side	2 choices; all on left, all on right
High speed taxi-way position	Automatic placement 2/3-3/4 of runway length
Runway width	150ft - 300ft in 50ft intervals

Table 1. (cont.)

Parameter	Specification
Runway length	4,500ft - 16,000ft in 100ft intervals
Displaced threshold	Offset between ALS and TO position
Runway marking types	2 short and long Piano keys selected by length - 8 types
Runway heading and identifier	Heading as station data, 36 runway numerals
Runway condition	4 types inc. dry, wet and snow
Visual landing aid type	8 types; 2-bar, 3-bar, VASI, PAPI, T-bar, etc.
Visual landing aid angle	10 steps; 2.4 - 3.4 degrees in 0.1 steps
Airfield gradient	-3.0 - +3.0 deg. in 0.01 deg. intervals

A wide range of other features are available within an AVGDB including:-

- Terrain containing cultural features (e.g. small villages around airfield)
- Taxi-way centre lines, centre lights, edge lights, stop bars and stop lights
- Runway tyre marks overlaid on painted markings
- Ground traffic perpendicular to runway at take-off end
- Docking gate on terminal at correct height for simulated aircraft
- Parked aircraft/vehicles at terminals
- Full weather effects
- Random star model
- Lighting intensity control as custom database
- Airport beacon with non-specific sequence
- VLAs and Wig-wags in modelled cans

- HAT (terrain height) for terrain and airfield surfaces

The database modelling workstation is used to pre-model the objects which make up an AVGDB. The objects are then assigned an animation in the database. When the host simulator has loaded the AVGDB, individual objects are selected by the host sending codes to activate the animation. The variants of the features listed in table 1 have individual parameters within the animations enabling each one to be selected as desired.

MVGDB

Scenario construction using a selection of generic library objects is provided by the Manual Variable Generic Database (MVGDB) facility resident on the SPACE visual system. The MVGDB scenario is known as 'InstantScene'. Objects modelled for the AVGDB are available for the object library used by InstantScene. This library is also enhanced by models selected from the full custom databases sharing a common texture wardrobe. These models will include major cultural and natural features in the environment as well as airfield buildings, runways and taxi-ways.

Creating Models Using the same modelling tool as custom databases, the initial MVGDB database is constructed off-line on the database modelling station , then it is exported to a PC resident utility known as 'DbBuilder' running under Microsoft windows to enable visualisation on the visual system. DbBuilder is one of a suite of utilities running on the PC termed the 'engineering console'. The engineering console can be either a lap-top PC situated in the cock-pit or a desk-top for off-board use. Using DbBuilder, library objects can be selected and positioned in the scene to enable rapid build and modification of the generic database.

Each operation carried out, i.e. position of selected object, will be stored in the event processor on the visual system when the database is transferred to it.

Recalling Models A reference code is assigned to enable subsequent selection by the host simulator. When the reference code is issued to the simulator, it invokes a download of the database (or objects into the database) and the visual system event processor automatically runs the commands that are stored to recreate the MVGDB as originally modelled.

Removing the Constraints

The components of Variable Generic Databases are effectively moving objects that can be placed and rotated at the request of the instructor. To enable complete flexibility in using the VGDB facility for training, the goal is to remove all modelling constraints, restrictions or special structures for controlling occlusion of hidden parts.

Achieving total freedom of placement of objects is realised by the provision of a full sub-pixel depth buffer as featured in the SPACE image generator. SPACE employs MultiSortingAlgorithm (MSA), a proprietary algorithm which includes a high precision depth comparison of 16 sub-samples made within each pixel before any anti-aliassing filter is applied. This provides for correct handling of interpenetrating and abutting polygons without the aliassing artefacts that would be present in systems with pixel level depth buffering only.

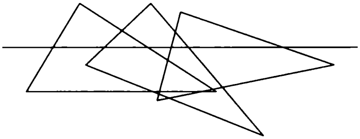


Figure 1. Interpenetrating Models for Rapid Terrain Generation

With this capability, basic terrain can be modelled rapidly from a few library models as illustrated in figure 1. Here, a pyramidal object is replicated a number of times to produce a set of rolling hills. All polygon intersections are handled correctly by the depth buffer architecture.

MSA also includes co-planar surface sort routines, a process for the overlaying of translucent surfaces and proximity algorithms.

Skip-Over Full sub-pixel depth buffering is expensive in rendering power so the SPACE system uses a special process known as 'skip-over' based on depth comparisons, to remove polygons which are hidden by large surfaces at a much lower screen resolution. This means that only the edges of polygons need to be handled at the sub-pixel level increasing the effective rendering throughput significantly.

A more detailed description of the architecture of the SPACE visual system can be found in a previously published paper[1].

Modelling Tools for InstantScenes

Two separate but complimentary interactive design tools are used for the creation of MVGDB InstantScenes; the 'IDEAL5' modelling tool based on Multi-Gen running on a Silicon Graphics workstation, and the 'DbBuilder' utility running on a PC under Microsoft windows.

The IDEAL5 Way

It is possible to create an InstantScene database from scratch with the DbBuilder tool, however, it is generally more efficient to generate the fundamentals off-line on IDEAL5. With IDEAL5 the user creates instances (copies) of individual library objects from the Global library then positions and configures them interactively as required. Off-line visualisation of the result can be made directly on the workstation, where all the features of wire-frame views, etc. are available.

The InstantScene file describing the database is then exported and read by DbBuilder which communicates it to the SPACE system for viewing. DbBuilder can then be used on-line for tuning, adjustments and additions as required.

Database Structure Although IDEAL5 provides many of the same mechanisms for modelling an InstantScene database as for modelling a standard one, InstantScenes are in fact different in nature, using a much reduced database structure.

An existing database may be converted to an InstantScene only if it contains this simple structure. Figure 2 shows the basic structure comprising just three primitives known as 'beads'. The Header bead has an 'Xfile' as its child. This has no function for InstantScene other than to act as a parent collector for the library beads. A library bead is created for each required object reference.

Selecting Objects The object library file containing the objects to be used is imported into IDEAL5. Each object instance that is required is referenced by an individual library bead. Associated with each library bead is an 'Object ID' which will be used by the SPACE system to identify each instance of the object. The Object ID will also be used to identify the object instance when using DbBuilder for subsequent tuning.

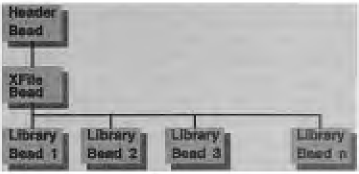


Figure 2. Database Structure For InstantScene

Actions and Evolutions Positioning and orientation of the referenced object is performed by applying a transformation to the parent library bead. Objects may be configured in various ways through the use of the SPACE animations facility. Each object is assigned 'Evolutions', linked to appropriate 'Actions', that specify the ways in which the object may be manipulated and configured.

Each Evolution specifies a set of values for whatever attribute is to be manipulated by the animation. these values are indexed by the phases in an Evolution cycle. Using this mechanism, an object can be configured according to any of the associated Evolution values defined for it, in a linear or stepwise manner. This is achieved by specifying which Evolution value the SPACE system should use in rendering an associated object.

Any of the Evolution types provided by SPACE may be employed in an InstantScene object to achieve a variety of results. Most notably, those listed in Table 2 giving the effects shown.

An instance of an object may be configured by specifying which Evolution value the SPACE system should use in rendering an associated object. This is achieved by specifying a 'Parameter Value' for each Action that controls some of the object's Evolutions. This Parameter Value specifies which phase in the associated Evolution's cycle is to affect the object.

Table 2. Evolution types available for InstantScene objects

Evolution	Function
Operationality	Select a specific version of a model from a set of variants e.g. a configuration of runway lights.
Visibility	Switch an object/part of an object on or off.
Material	Switch between different materials, e.g. winter or summer textures and colours.
Position	Position an object/part of an object, e.g. change the size of a runway by moving the different sections.

For example, a runway is required to have a set of different runway numbers selectable. Part of the runway object would reference an Operationality evolution to allow the different runway numbers to be selected. Each number would be an individual sub-object beneath a controlling Selector bead that references the Evolution and the Evolution indexes each sub-object in turn. The database structure would typically be as shown in Figure 3.

Exporting an InstantScene for Visualisation When an InstantScene database is ready to be visualised on the SPACE system, it is exported in a special format, unlike the standard format used for conventional databases. The file is written in a format suitable for import by DbBuilder which records all data regarding object selection and positioning, along with the Parameter Values used to configure each object instance. Using this file, DbBuilder will request creation of the database by the real-time on the visual system and may be used to edit it if necessary.

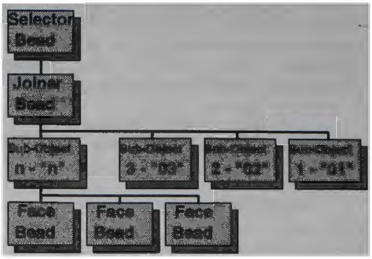


Figure 3. InstantScene Structure For Runway Number Selection

The DbBuilder Utility

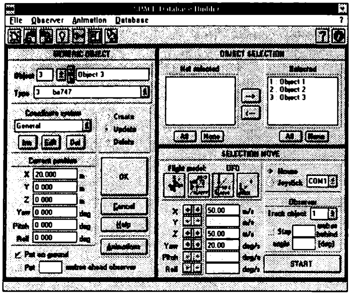
This utility gives the user a method of creating and editing MVGDB InstantScenes from the PC, using the SPACE visual system to visualise the result in real-time. The principle is to place on the database, objects taken from an object library including runways, buildings, lighting systems etc., with the help of the PC windows interface.

DbBuilder has the capability to create InstantScene databases from scratch but in general, will only be used for tuning and adjustments to databases which have been generated using IDEAL5 as described above. The utility has a user friendly interface via Microsoft windows together with extensive on-line help facilities.

With DbBuilder, a number of features are available. Objects may be instanced from the Global library in a similar way to IDEAL5, positioned and oriented as desired. A number of objects may be bound together to form a group which can then be moved en-bloc to a new location.

Another feature is the ability to instance animations (Evolutions and Actions). These are used to provide selectable parts of an object such as different runway numbers. Also, via animations, different variants of objects can be provided such as a B767 aircraft with the option of wheels/undercarriage and/or a Marshaller. The wheel animation is set for take-off, landing and taxi-ing but not for flight, while the Marshaller will only be selected for docking operations.

Figure 4 shows the main user interface window of DbBuilder from which the operations described here are performed.



Object 3 Object 3
Type 3 ba747
General
1 : Object 1
2 : Object 2
3 : Object 3
X 20.000 m
Y 0.000 m
Z 0.000 m
Yaw 0.000 deg
Pitch 0.000 deg
Roll 0.000 deg
COM1
Track object 1
X 50.00 m/s
Y m/s
Z 50.00 m/s
Yaw 20.00 deg/s
Pitch deg/s
Roll deg/s

Figure 4. DbBuilder User Interface

The basic steps in using DbBuilder are as follows:-

- Connect to the SPACE visual system specifying either a new database or existing one for edit
- Select objects to be used in the database
- Move objects in real-time i.e. whilst viewing on the visual system
- Select animations to be activated on objects
- Attach objects to co-ordinate systems for grouping
- Generate the database on the SPACE system

Visual System Connection A scenario number is specified, when connecting to the visual system, which is associated to the event context file stored on the visual system disk. This event context file is then only usable with this scenario number. Once connected, the PC copies the scenario file from the visual, reads it, and extracts the name of the object library. The PC then copies the corresponding object file from the visual disk in order to retrieve the list of available objects.

Object Versatility DbBuilder has many functions available for selection and manipulation of objects, :-

- Create an object
- Select an object, several objects or all objects (when selected, objects blink on the visual display)
- Cut / copy / paste objects
- Group / ungroup objects
- Move a selection - freeze on axis, move relative or absolute position, align on grid, translation or rotation.

When all the objects in the database have been placed in the required positions, the InstantScene database is transferred to be generated on the visual system. This enables the host simulator to instanciate the on-line database with objects and animations defined and placed by DbBuilder and will be loaded as the database used during a training exercise, in real-time. The host does not need to consider any other control of the visual system.

Several on-line databases can be created on the visual system from a database on the PC. When generating a database on the visual disk, a number is specified which will determine the destination filename of the form CNTX_YYY_XXX where YYY is a base scenario number and XXX is the unique context number for this particular InstantScene database. These filename identifiers become the reference codes which shall be used by the host computer when calling up (replaying) the database for training use.

All major visual parameters can also be changed in real-time by DbBuilder to assist viewing databases in different ambient conditions. Parameters controllable are:-

- Visibility distance (metre):
  eye-point fog range
- Ambient light (candela/m^2):
  global light level
- Directive light (candela/m^2):
  sun light source, depending on time of day
- Observer position (via keyboard, mouse or joystick)

Overview of the Complete Process

Interaction between the two design tool components; IDEAL5 and DbBuilder, and with the visual real-time system, is illustrated in Figure 5. The main interconnect functions and operations are described below.

(1) Objects created on IDEAL5 are added to the global library on the visual system.

(2) The initial InstantScene database created by IDEAL5 is sent to DbBuilder on the PC via a filename *.dbc. The *.dbc file is basically, a list of object instances with animation data. Animation data consists of two sets; default per object type, and active per object instance. DbBuilder uses the file to visualise the database.

(3) DbBuilder connects to the visual host via ethernet. When the database is requested of the visual system, an event file is generated by the real-time, recording all object instance positions and animations, and becomes the unique scenario file enabling selection by the host simulator.

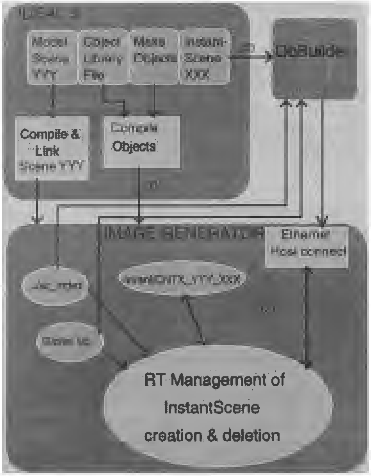


Figure 5. MVGDB Overal Design System

Applications

Uses for AVGDB and MVGDB span the whole range of visual simulation applications. In the civil arena, VGDBs are primarily aimed at secondary diversion airfields where full detail is not usually required as described earlier in the paper. However, training operators are also beginning to use the feature for some of the mainstream training airfields within the core library because it provides a very cost effective method to build up the library.

In fact, the level of detail and match with the airfield can be quite high especially considering that the runway and approach lighting system, which is the most important part of the airfield, will be as accurate as any full custom database.

Mission rehearsal in military applications is another area of application ideally suited to VGDB use. The very nature of how conflict situations evolve means that missions often can not be planned more than hours in advance. In these circumstances, it is necessary to rapidly provide scenarios covering areas of unfamiliar enemy territory to enable training on demand for pilots in conflict situations. This is equally true for tank crews (drivers, gunners, etc.) as it is for combat aircraft pilots.

Conclusion

The variable generic database creation tools have been designed with the user in mind, and the use of windows interfaces on the PC have enabled familiarity to be built in. This helps to accelerate the learning process, increasing usage and thus providing optimum pay-back from the feature.

Variable Generic Databases produced by the tools described in this paper are in use now by a number of civil airline customers and the feedback from these users is already very positive. As more civil and military applications arise, the feedback from users shall be used to enable continued evolution and further improvements to be made to the tools.

An option currently under investigation is to provide IDEAL5 with the ability to directly create the real-time event file on the image generator thus bypassing the DbBuilder tools.

This will add further flexibility by enabling direct visualisation of an InstantScene on the visual system when the engineering console is occupied for other operational use. Subsequent editing of the InstantScene database will still be possible with DbBuilder if required.

The fundamental simplicity of approach by AVGDB and MVGDB are made possible only by the availability in the image generator of a full sub-pixel high resolution depth buffer as described in the paper. Without this, the constraints on placement of objects and increased object library incurred would reduce the flexibility, and therefore the ability to produce very low cost real training scenarios.

Continued refinement to the tools is ongoing now, and will help to ensure that the pay-back in training value will continue to increase relative to the investment in tools and modelling time.

REFERENCES

1. Jarvis, K. M., "Shifting the Visual Paradigm", Proceedings of the IMAGE VII Conference (pp. 67-75), Tuscon, Arizona, June 1994.

AIAA-96-3546-CP

ON HIERARCHICAL MULTIRESOLUTION TERRAIN POLYGON GENERATION USING WAVELET TRANSFORMS AND OPTIMAL MESHING

Robert W. Fuentes, Donald E. McArthur, and Venkat Devarajan
PO Box 19016, Department of Electrical Engineering
The University of Texas at Arlington
Arlington, Tx 76109
USA

Abstract

In this paper we investigate an approach for generating a hierarchical, multiresolution polygonal data base from raw elevation data using the wavelet transform. This is based on our earlier experiments in the use of the wavelet transform to produce progressively lower resolution data from a high resolution data set. Our approach is to utilize wavelet transforms and optimal mesh generation techniques to create an initial coarse mesh and determine the significant vertices that need to be inserted into the mesh. This methodology retains the dominant terrain features at each level and maintains a database hierarchy. Results of this procedure are presented with simulated elevation data. Timelines to generate the database and the automation features are highlighted.

Introduction

Visual systems and the visual perception they produce are an significant part of a flight simulation or mission rehearsal system. It is critical that the series of scenes are processed in an efficient manner and that the representation is true to the data being modeled. In both cases the model must be free of artifacts which might produce conflicting stimulus, resulting in disorientation and possibly motion sickness, in particular for mission rehearsal the user must feel confident in the model presented

The two components of a visual system are the data base generation system and the image generation (IG) system. In the data base system, starting with input source data such as US Defense Mapping Agency's digital terrain elevation data (DTED) of the gaming area, texture images corresponding to the DTED, and model data of the objects to be placed in the gaming area, are organized so that a data base is created. This data base is then loaded on to the IG which uses it to drape realistic texture onto a terrain skin of the visible portion of the gaming area at real time rates. A high resolution representation of terrain skins constructed from polygons (usually triangles) is essential for realistic rendering of scenes. The technical challenge is keyed from the need to represent the terrain with the greatest possible geometric accuracy while performing a trade-off to utilize the least number of polygons. In particular, navigable multiple levels of detail (LOD) of the terrain and model data bases must be created as accurately and automatically as possible.

Problem Statement

The scope of the related problem addressed by this paper is the automated generation of hierarchically structured multiple LOD data bases from the raw DTED. Support for critical features and critical vertices must be provided. For data base systems that feed state-of-the art IG's, continuous level of detail is generated in real time and there is usually no time to perform filtering at all. In this paper, we describe the results of our effort to automatically generate terrain LODs from a DTED input using the wavelet transform and mesh optimization. The problem at this level is to provide an automated process for triangulating the DTED at a higher density where there are more critical features and a lower density where there are little or no critical features. The number of terrain polygons must be reduced to a minimum and accuracy of representation of the terrain skin must be maximized.

Several issues regarding conflicting requirements between high quality image generation and real time processing have been previously presented [4]. There are many applications for eyepoint specific models, where fine details are included as required for a specific viewpoint position. (i.e., the accuracy you can tolerate for terrain representation is directly related to the particular eyepoint position of the observer).

The identified need for a hierarchical model and subsequent benefits include the representation of the terrain at a virtually continuous range of resolution, support of point location queries, structure navigation to support panning and zooming, and variable resolution rendering [2].

For real-time applications it is necessary to quickly insert and delete high resolution patches thereby performing incremental updates which would increase resolution when required and decrease resolution when permissible under resolution requirements. Seamless merging of high resolution & low resolution is also required so that the rendered scene provides a natural representation of the terrain free of holes and surface discontinuities.

Previous Work

The importance of critical features and some methods of extraction have been presented by Scarlatos et al [10, 11]. These methods are spatial domain based and somewhat ad hoc in that heuristic rules are used in deciding what a critical feature is.

For a good introduction to the wavelets and wavelet transforms see references [1,9]. A tutorial on the use of wavelet transforms and their applications to computer graphics can be found in [12] in which the relevant discussion is the multiresolution analysis using the wavelet transform. Multiresolution in this context is the same as the multiple level of detail generation. The referenced tutorial paper is restricted to the continuous wavelet basis function whereas the generation of polygons for rendering requires the discrete basis function.

Many key issues regarding triangulation via hierarchical subdivision and multiresolutional surface modeling were presented by De Floriani and Puppo [2]. Their paper describes a well-founded process based on nested triangulations, providing multiresolution surface models, along with a surface model referred to as a "Hierarchical Triangulated Surface". In their concluding remarks they identify a common problem in triangulated representations, elongated or slivery triangles introduced as the number of degrees of resolution becomes larger.

Closely related work for multiresolution surface generation based on local spectral estimates of the surface through wavelet representation was accomplished in reference [5]. The position of their paper differs in that a bottom up approach was proposed (i.e., removal of unimportant vertices) and the apparent lack of hierarchical based triangulation.

Discrete Wavelet Transform Summary

The discrete wavelet transformation (DWT) is a transformation which has several key properties related to surface approximation and reconstruction. Of particular importance to our approach is that the DWT has a vanishing first moment, i.e.,

$$\Sigma(-1)^k k a_k = 0;(k \in \{0,\ldots 2N-1\})$$

The wavelet coefficients used for our processing are the Daubechies class, specifically Daubechies-6 [7]. These were chosen in part due to their provision for highly localized analysis. The following reference provides additional detail on the DWT process [6], while [3] describes wavelets for the specific application to LOD generation.

A two dimensional wavelet transformation can be accomplished by first applying the low frequency and high frequency wavelet filter transformation along each row of the data set independently to produce a low and high set of wavelet coefficients. Then the wavelet filter transformation is applied along each column of the low and high data coefficients independently to produce a low-low (LL), low-high (LH), high-low (HL), and high-high (HH) set of data coefficients. The LL set is the smoothed representation of the data, while the LH, HL, and HH are the details (i.e., abrupt changes in the data). The LL can then be transformed again to extract additional details (LH2, HL2, HH2, from LL1; where the number identifies the level of the transformation, reference Figure 1) and produces a set of data coefficients corresponding to smoother data representation. A coarse version of the original data can be obtained by performing a reverse wavelet process while discarding all high frequency data coefficients.

Generation of a Hierarchical Set of Triangles

The approach utilizes the wavelet transform to create a very coarse approximation of the original data (reconstructed from LL4 data coefficients) as a starting point for the surface model and utilizes the less smooth approximations (reconstructed from LL3, LL2, and LL1 data coefficients) as a control for the identification of which vertices to add to the model to provide a higher resolution representation. Figure 2 provides a block diagram of this approach.

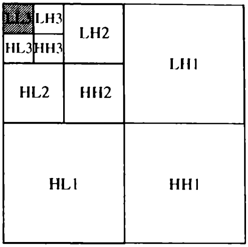


Figure 1 Level 3 Wavelet Data Coefficients

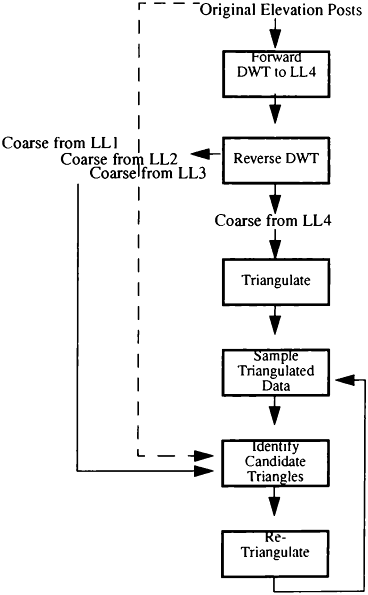


Figure 2 Proposed Approach

A perspective view of the original elevation posts is provided in Figure 9. The outlined process, produces a "root" model (Level 0) from the triangulate operation performed on the "Coarse from LL4" data, the triangulation is provided in Figure 3 and the perspective view in Figure 10. The proposed triangulation is a top-down approach which is initiated with two triangles formed by placing a diagonal across the four corner posts. The maximum error of each triangulation is compared to the defined tolerance to determine whether the triangle must be further subdivided to produce the required resolution. The basic subdivision of a triangle is accomplished by interpolating the elevation of the midpoints of its sides and then dividing it into four smaller triangles by joining these midpoints. It is apparent that with no further processing, this method will produce a continuous surface model for cases where a triangle on one side of an edge is subdivided. However, in the case that triangles on both sides of an edge are subdivided, the interpolated value must be replaced by the actual elevation of the post in the data set being evaluated. In this way, the neighboring edges are represented in a continuous fashion.

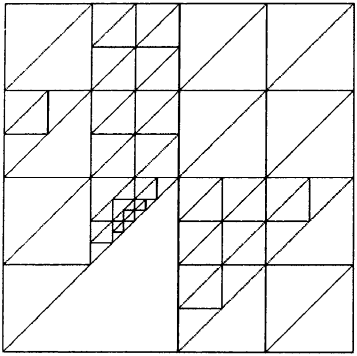


Figure 3 Triangulation of Root (Level 0)

This root model is then sampled and compared to the next coarsest representation from the reverse DWT, "Coarse from LL3" to identify which triangles must be broken down further in the next set of processing. The resulting triangulation is provided in Figure 4.

This process is then repeated using better approximations to the original data and eventually the original data itself to create a progressively finer representation of the data. Figures 5, 6, and 7 provide the hierarchically derived results from these steps, while Figures 11 and 12 provide corresponding perspective views for the Level 2 and Level 4 data models.

Table 1, identifies the results of the experimental processing. It includes tolerances used for each round of processing along with the number of triangles generated, and the RMS error from the original data set. The tolerance was used in comparing the input to the comparator. This may be a DWT reconstruction from the smoothed coefficients (LL3, LL2, LL1) or in the final case, the original data itself. The comparator provides orderly control for the further breakdown of triangles and subsequent increase in model resolution. A decrease in the tolerance is not always necessary in order to produce additional triangulation, since the

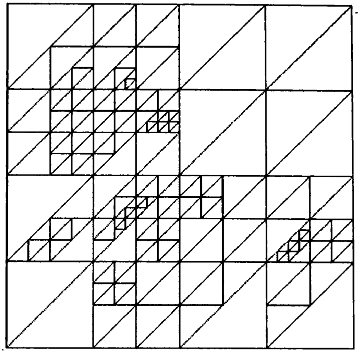


Figure 4 Triangulation of Level 1

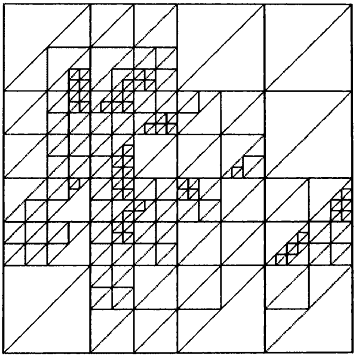


Figure 5 Triangulation of Level 2

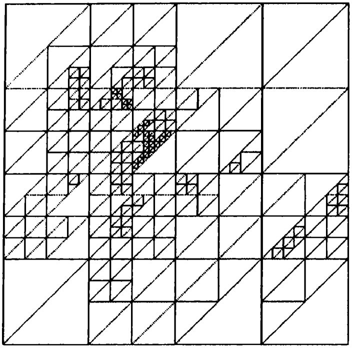


Figure 6 Triangulation of Level 3

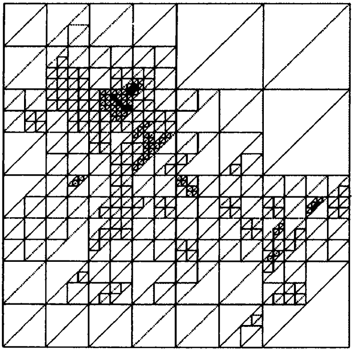


Figure 7 Triangulation of Level 4

comparator is different at each level (reference the last case in Table 1).

Table 1: Triangulation Results

Input	Comp	Output	Tol	Nt	RMS
Coarse from LL4	N/A	Root	200	77	70.47
Root	Coarse from LL3	Level 1	150	194	54.76
Level 1	Coarse from LL2	Level 2	100	302	44.10
Level 2	Coarse from LL1	Level 3	25	368	42.74
Level 3	Original	Level 4	100	728	31.30
Comp => comparator Tol => tolerance Nt => number of triangles RMS is the error from the original data set					

Typical values of the low and high data coefficients along with representative displays of the differences between successive levels of the wavelet transform were previously provided [3].

The hierarchical structure of the triangulations is apparent from the outputs displayed in Figures 3 -7. The flat areas of the original data set are maintained with a few large triangles, while the areas where peaks and valleys dominate are initially displayed with large triangles, but progressive resolution is provided in subsequent levels.

The benefit of utilizing the reversed DWT to create the root model, instead of the original raw data is that the wavelet transformed approximation provides a starting point that is representative of the data, but requires a fewer number of polygons. Additionally, it does not introduce areas of high detail too soon in the process, in which case it allows the detail to be gradually introduced. This is illustrated by the data shown in Figure 8 which shows a triangulation derived from the original data, using the same tolerance as the proposed root model. It results in nearly twice as many triangles (134, compared to 77) as our proposed root, but only provides a slightly better RMS error result (67.93, compared to 70.47) and tends to introduce higher detail sooner.

Wavelet Transforms of the Model

The vanishing of the first moment of the chosen wavelet transform predicts that the DWT of any planar patch should be equal to zero, as long as it is not on a boundary edge. This point is critical in the application of wavelets to multiresolution surface modeling and is illustrated by comparing the root triangulation of Figure 3, with the display of wavelet coefficients for the root in Figures 13 and 14. The first is the HH1 set of

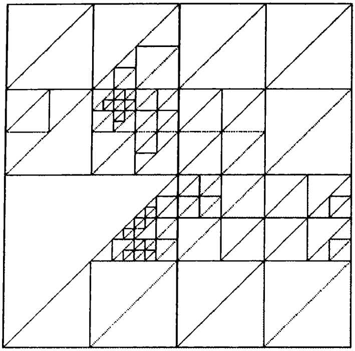


Figure 8 Triangulation of Original Data

coefficients and the second is the LH1 coefficients. In the first case, the diagonal components of the detail are highlighted, while in the second the details along the vertical are more prominent. But more importantly, in both cases the areas of the spatial data set which are relatively flat correspond to zero high frequency values in the wavelet domain.

Another point of interest concerning the wavelet analysis can be identified by comparing the LL4 wavelet coefficients of the original data set to LL4 of the root model. Essentially, this is a wavelet coefficient description of the accuracy of the model. In referencing Figures 15 and 16 respectively, it can be seen that the two are qualitatively similar.

Conclusions

In this paper we have presented a discussion of our experiments and the corresponding results for our approach to generating a hierarchical multiresolution terrain data base utilizing discrete wavelet transforms and mesh optimization. First the DWT was used to create a coarse approximation of the raw elevation posts. A triangulated mesh was then created from these new post values, forming the root model. Successively higher resolution was then added by comparing the current model with a less coarse wavelet generated representation of the original data, to guide the insertion of vertices for critical points. Essentially, the triangulation is maintained by simply refining (i.e., retriangulating) the triangles which have a maximum error greater than the defined tolerance. This produces the next step in a hierarchical triangulation and this process is then repeated using less coarse wavelet generated representations to build a complete multiresolution LOD data base. The current hierarchical mesh is optimized only in one dimension (i.e., modifications are made to the z-values of posts), while the two dimensional positioning of the posts is not addressed.

The top-down approach proposed creates.a continuous surface model at each level, which eliminates holes or cracks in the model. This is accomplished by replacing interpolated elevation values with actual data set values when an edge is split on both sides. This feature is required for surfaces built for the proposed applications.

Our method produces a set of triangles which can be used to construct an eyepoint specific representation, by piecing together areas of varying resolution to form a model with the number of triangles utilized directly related to the detail required.

The work done in implementing our approach was done on a PC (Intel P-90, with 32M of RAM) in Object Oriented C++. The code is not quite optimized, however the timing required to process a set of elevation posts with dimension 128 by 128 is in the range of 15 seconds to import the data cell, forward transform it four levels, reverse transform it to create the coarse approximation, and output all wavelet data coefficients to file. The process of creating the root model and the subsequent Level 1, Level 2, Level 3, and Level 4 models takes on the order of one to two minutes.

Future Work

Future work must be accomplished to allow more flexibility in the triangulation process. The mesh optimization should be refined to allow for spatial repositioning of the interior elevation posts to provide a model that generates fewer triangles for similar results.

Based on our previous research and this current effort, we are convinced that more use of wavelet coefficients can be made as a guide for optimization . Analysis in the wavelet coefficient domain looks promising in that the zero moment feature of the wavelet transform can provide some useful information in further optimizing this process.

References

[1] I. Daubechies, "Orthonormal Bases of Compactly Supported Wavelets", Comm, Pure and Applied Math. 41, pp. 909-996 (1988)

[2] L. De Floriani and E. Puppo, "Hierarchical Triangulation for Multiresolution Surface Description", ACM Transactions on Graphics, 14(4), ACM Press, 1995, pp. 363-411.

[3] V. Devarajan, R. Fuentes, and D. E. McArthur, "An Approach to Multiple Levels of Detail Generation from Digital Terrain Elevation Data Using Wavelet Transforms", 7th International Training Equipment Conference Proc. '96, pp. 255-262.

[4] V. Devarajan and D. E. McArthur, "Terrain Modeling for Real-Time Photo-Texture Based Visual Simulation", American Society for Photogrammetry & Remote Sensing / American Congress on Surveying and Mapping '93, Vol. 1, pp. 129-138.

[5] M. H. Gross, R. Gatti, and O. Staadt, "Fast Multiresolution Surface Meshing", Proc. of the IEEE Visualization '95, pp. 135-142, IEEE Computer Society Press, 1995.

[6] A. L. Graps, "An Introduction to Wavelets", IEEE Computational Sciences and Engineering, Vol. 2, No. 2, Summer 1995, pp. 50-61.

[7] M. L. Hilton, B. D. Jawerth and A. Sengupta, "Compressing Still and Moving Images with Wavelets",

Multimedia Systems, Springer Verlag, 1994, pp. 218-227.

[8]H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle, "Mesh Optimization", Computer Graphics (SIGGRAPH '93 Proceedings) pp. 19-26.

[9] S. Mallat, "A Theory of Multiresolution Signal Decomposition - the Wavelet Representation", IEEE - PAMI, PAMI-11, pp. 674-696 (1989)

[10] L. L. Scarlatos, "An Automated Critical Line Detector for Digital Elevation Matrices", Proc. of the 1990 ASPRS/ACSM Annual Convention, pp. 115-122.

[11] L. L. Scarlatos, "A Compact Terrain Model Based on Critical Topographic Features", Proc. of Auto-Carto 9, pp. 146-155.

[12] E. J. Stollnitz, T. D. DeRose and D. H. Salesin, "Wavelets for Computer Graphics: A Primer Part 2", IEEE Computer Graphics and Applications, July 1995, pp. 75-85.

[13] W. R. Zettler, J. Huffman and D. C. P. Linden, "Application of Compactly Supported Wavelets to Image Compression", Algorithms and Techniques, Proc. of APIE Vol. 1244, pp. 150-160.

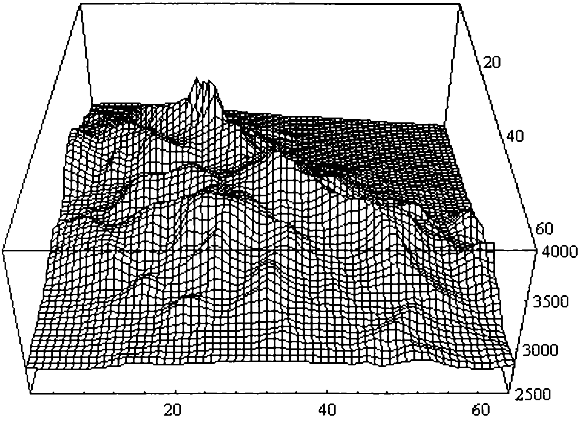


Figure 9 Perspective - Original Data

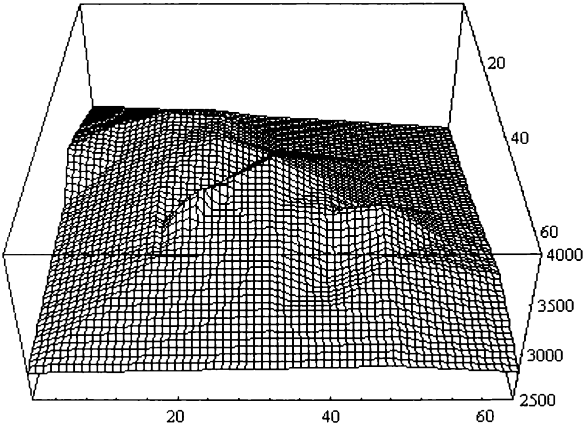


Figure 10 Perspective - Root (Level 0)

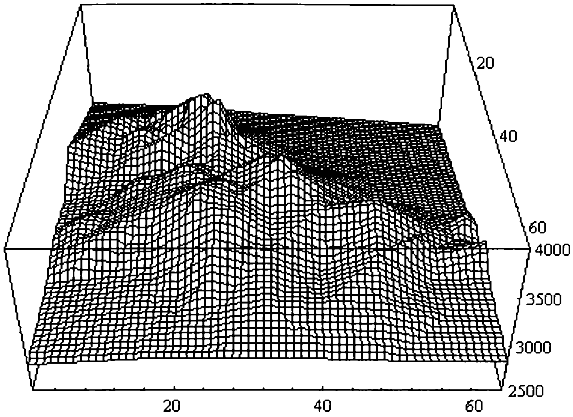


Figure 11 Perspective Level - 2

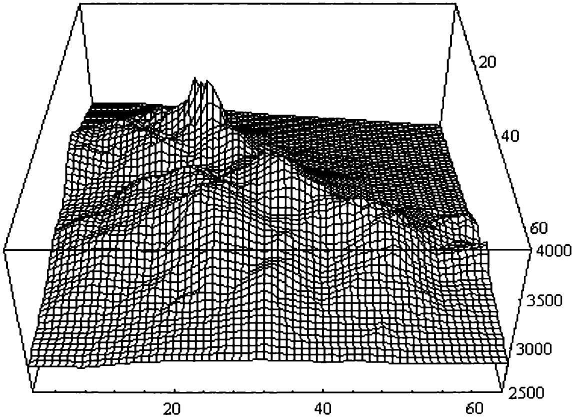


Figure 12 Perspective - Level 4

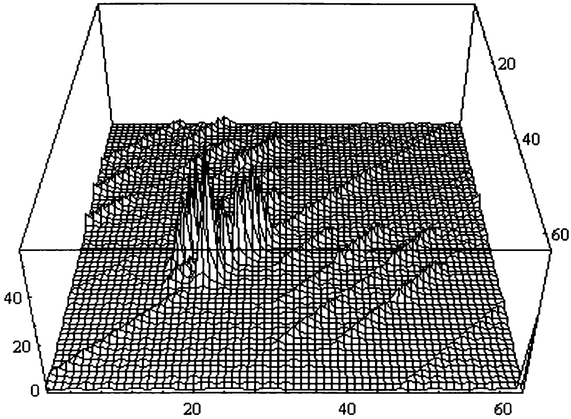


Figure 13 Coarse From LL4 - HH1 Wavelet Data Coefficients

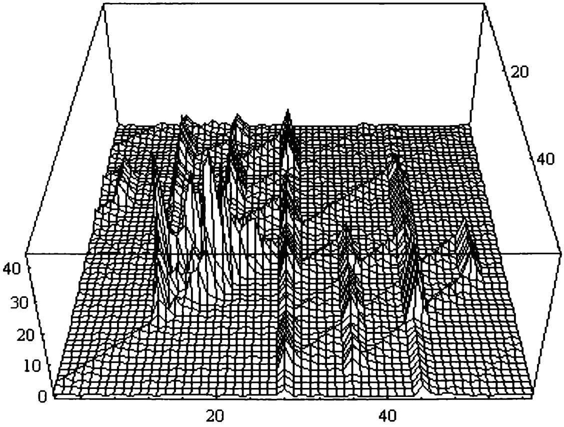


Figure 14 Coarse From LL4 - HL1 Wavelet Data Coefficients

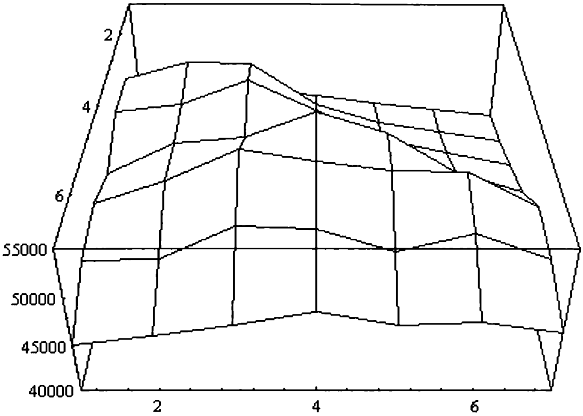


Figure 15 Original - LL4 Wavelet Coefficients

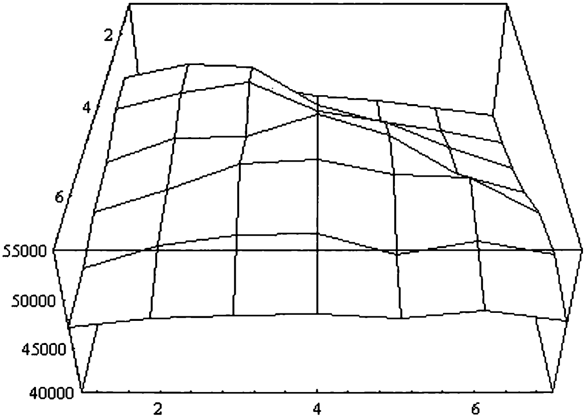


Figure 16 Root - LL4 Wavelet Coefficients

AIAA-96-3547-CP

A Graphics User Interface for Low-Cost Flight Simulation Software Using Graphical Programming

Jari Vilenius* and Pasi Kemppainen†
Helsinki University of Technology, Laboratory of Aerodynamics
Sahkomiehentie 4, FIN–02150, Espoo, FINLAND

This paper describes the structure of a graphics user interface (GUI) for a low-cost PC-based flight simulation software (HUTFLY). The GUI was created using LabVIEW software, which is based on a graphical programming language, G. No lines of conventional source code were written. This study proved that graphical programming offers an affordable alternative to conventional text-based programming. Visually pleasant interfaces can be created without an extensive knowledge of graphics programming, which speeds up the development process significantly. Graphical programming also makes it possible to modify the existing interface with minimal efforts. Provided that the graphical presentation of the instruments shown in this paper is deemed acceptable, this approach can be highly recommended.

1 Introduction

Due to their relatively high performance-price ratio, PCs have become more and more popular in low-cost flight simulators. The entertainment industry has taken full advantage of the graphics capabilities of a modern PC and built visually breathtaking flight simulators, usually at the cost of mathematical modelling of the aircraft.

There are engineering applications where a visual display with moving scene is not necessary and pseudo-real-time execution is acceptable, but no compromises can be made over the mathematical modelling of the aircraft and environmental effects. However, an easy-to-use interface and efficient post-processing capabilities are necessary for increasing the effectiveness of the research. The graphics user interface and the core of the flight simulation code must also be flexible to allow necessary modifications to be made without great effort.

The core of HUTFLY has been written in Fortran. The graphics capabilities of Fortran are, however, limited, which makes it inefficient for graphics-oriented applications. On the other hand, it takes quite a lot of time to get acquainted with the details of graphics programming. Therefore, the objective of this study was to find tools that could make the development process of the interface faster and more efficient compared to conventional text-based programming. After having analyzed the tools available on the market, LabVIEW[‡] software was selected for the study. A reliable user support, popularity, unique development environment and positive experiences gathered by our wind-tunnel group made the choice quite easy.

2 Current flight simulation software

HUTFLY is a Fortran-based flight simulation software, which has been developed at Helsinki University of Technology in the Laboratory of Aerodynamics. It runs on a PC, and it is suitable for engineering analysis and education. The methods implemented in HUTFLY have worked well in flight simulation applications over the years. HUTFLY has been applied to flight control system and optimal trajectory studies. Malfunctions, external store separations and wind effects can also be simulated, and the flight path of a real aircraft can be reconstructed by feeding recorded control surface deflections into the simulation. HUTFLY can also be used for demonstrating the principles of flight mechanics and aerodynamics.

The structure of the software has been kept modular to facilitate the development and debugging processes. Due to modularity, new modules can be added without significant efforts. The basic modules are presented in Figure 1. The trajectory of the aircraft is determined us-

[*] Research Engineer, Member AIAA
e-mail: jvileniu@hermes.hut.fi

[†] Research Engineer,
e-mail: pkemppai@hermes.hut.fi

[‡] LabVIEW is a trademark of National Instruments Corporation

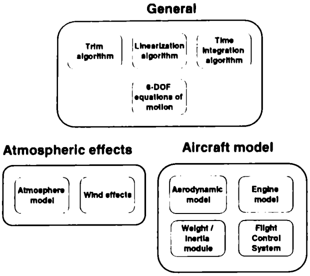


Fig. 1: The basic modules of HUTFLY flight simulation software.

ing the 6-DOF equations of motion with a flat-earth approximation, and the differential equations are solved numerically using an Euler or a fourth-order Runge-Kutta method. A trim algorithm is also available. The trim (steady-state) condition can be employed as an initial state for the simulation. The trim algorithm is based on the minimization of a cost function using the Simplex method. The cost function is the sum of the squares of certain state derivatives. It is also possible to build a linearized aircraft model at a chosen trim condition. The trim and linearization algorithms are modified versions of the ones presented in Stevens.[1] HUTFLY has a wind module which includes models for steady wind, gusts and atmospheric turbulence. Gusts are modelled using (1-cos) approximation and turbulence is described using a Dryden model. Calibrated airspeed (CAS) and the properties of the International Standard Atmosphere (ISA) and associated off-standard atmospheres are calculated in an atmosphere module.

The characteristics of the airframe, engine and an automatic flight control system are described in an individual module, which makes it possible to incorporate new models of varying complexity into the simulation with minimum effort. At the moment there are two complete aircraft models available: The F-16 with F100-PW-100 afterburning turbofan and BAe Hawk Mk.51 with Rolls-Royce/Turboméca Adour Mk 851 turbofan. The former was received from the Flight Research Laboratory of Kansas University and the latter was extracted from the BAe Hawk training simulator located at the Finnish Air Force Academy.

3 LabVIEW software

LabVIEW[2] (LV) is a program development application. It differs, however, from traditional program development applications like C or Fortran in one important respect. Conventional programming languages are text-based, which means that applications are created by writing a block of code. The block is then compiled and linked to the other blocks. LV is based on a graphical programming language, G, and the programs are created in a block diagram form. Programming problems are solved by expressing the relations and actions using graphical symbols, so only a little programming experience is needed. LV has efficient program development tools to make debugging easier. Breakpoints can be set into the program, single steps can be taken through the program and the execution can be animated to check how the data is passed through the program. LV also has many libraries of functions and subroutines for most programming tasks. In addition, a lot of application-specific libraries for data acquisition, data analysis, data presentation, as well as data storage and serial instrument control are included.

LV programs are called virtual instruments (VIs) because their appearance and operation imitate actual instruments, like meters or analyzers. VIs are analogous to the functions created by the conventional programming languages. VIs have both an interactive user interface and a source code equivalent, and they accept parameters from higher level VIs.

The interactive user interface is called front panel because it simulates the panel of the physical instrument. Knobs, push-buttons, graphs and other controls and indicators can be included in the front panel. Each VI receives instructions from the block diagram which describes the solution to the programming problem in a graphical form. The block diagram is constructed in G. An example of a simple front panel with associated block diagram is presented in Figure 2. VIs are hierarchical and modular so it is possible to create a VI inside another VI (subVI). Thus, the programming problem can be divided into several tasks, which can further be divided into subtasks until the problem is built up from simple subtasks. A low-level VI can be a solution to one subtask. Several low-level VIs can be connected to a larger level VI. A top-level VI represents a solution to the problem in a general form. Due to modularity the efficiency of the development process increases, and since the VI can be executed by itself, the debugging becomes much easier.

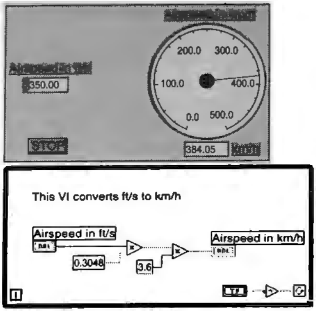
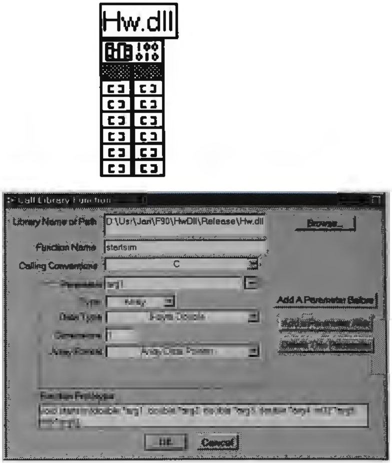


Fig. 2: A simple front panel and associated block diagram.

4 The Graphics User Interface (GUI)

The communication between LV and Fortran flight simulation software is effected through dynamic link libraries (DLL).[3,4] DLLs contain an executable code and they can be called either by an application or another DLL. A Fortran DLL can, however, only provide computational services or perform file I/O. It cannot perform screen I/O. DLLs and static libraries have many common features. However, the way they are linked to an application is different. DLLs are linked "dynamically" at run-time while static libraries are linked when the application is linked to form an executable file. DLLs can be used for building mixed language applications or they can be linked to Windows applications that can call DLLs, such as LV or MS Excel for Windows.

DLLs are called from LV using a "Call Library Function" which is shown in Figure 3. Before calling a DLL, the user has to define a calling convention for the DLL to make sure that LV and DLL use the same naming convention and argument-passing protocol. If the calling convention is defined incorrectly, the communication between LV and DLL will not work correctly, which may lead to a system crash. The schematic structure of the communication between LV and the flight simulation software DLL is presented in Figure 4. Figure 6 presents an overview of the simulation session. The simulation session starts from the main menu, which is presented in Figure 7. The trim button opens the trim window (Figure 8), where the user defines the required trim condition by giving Mach number, climb angle, altitude and turn rate. The user also has to define, how many iterations the trim algorithm will perform and how accurately a throttle position is calculated. The trim results are displayed in a result window, and they can also be stored in the trim file. The linearize button opens a linearization window (Figure 9), where the user defines the trim file, subset of states, controls and outputs, which are used as a basis of the linearized aircraft model. The results of linearization are shown in a separate window, and they can be stored in the result file. The simulate button opens a cockpit display (Figure 10), which has the same basic displays as its real counterpart. The cockpit display is also equipped with an artificial horizon and a warning panel. The artificial horizon was created by taking advantage of LV's



Library Name or Path: D:\Usr\Jari\F90\HwDll\Release\Hw.dll

Function Name: startsim

Calling Conventions: C

Parameter: arg1

Type: Array

Data Type: 8-byte Double

Dimensions: 1

Array Format: Array Data Pointer

Function Prototype: void startsim(double *arg1, double *arg2, double *arg3, double *arg4, int32 *arg5, [illegible] *arg6);

Fig. 3: The structure of the Call Library Function.

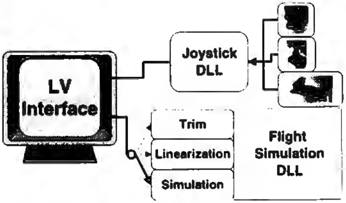
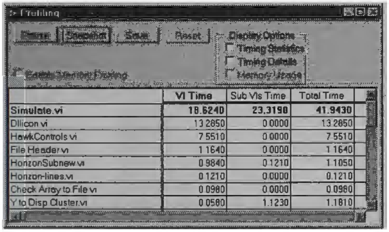


Fig. 4: Communication between LabVIEW (LV) and Dynamic Link Libraries (DLL).

versatile XY-graph tool. The warning panel was built using LV's boolean buttons. There are warnings for low altitude and approaching stall. A warning light will also illuminate, when rate or deflection limits of control surfaces have been reached. This feature was found useful especially in a flight control system design.

The aircraft can be controlled during the simulation either by a joystick-throttle-pedal combination or by feeding controls from an input file. The user will get information about the attitude, position and the velocity of the aircraft in graphical form. In addition, stored results can be visualized using graphics tools that are included in LV, or visualization can be provided independently using any commercial visualization package.

HUTFLY was tested under Windows 95 using a PC equipped with a 150 MHz Pentium processor. The execution speed is dependent on the integration time step, refresh rate of the screen and the number of the graphic displays in the GUI. XY graphs require much more resources compared to numerical indicators like meters or gauges. This was proved in the test, where the artificial horizon took 50-90% of the time that was needed for updating the display. Times were measured using LV's performance profiler tool (Figure 5). During the test runs, a pseudo-real-time execution was achieved using an Euler time integration algorithm with the following settings: The integration time step was 0.002 seconds in the flight simulation DLL and the refresh rate of the display was 5Hz. The resulting jerkiness was not considered excessively disturbing.



	VI Time	Sub VIs Time	Total Time
Simulate.vi	18.6240	23.3190	41.9430
Dllicon.vi	13.2850	0.0000	13.2850
HawkControls.vi	7.5510	0.0000	7.5510
File Header.vi	1.1640	0.0000	1.1640
HorizonSubnew.vi	0.9840	0.1210	1.1050
Horizon-lines.vi	0.1210	0.0000	0.1210
Check Array to File.vi	0.0980	0.0000	0.0980
Y to Disp Cluster.vi	0.0580	1.1230	1.1810

Fig. 5: LV performance profiler tool.

Some details of the source code are presented in Figures 11-12. Many basic elements (loops, logical expressions, arithmetical operations) have been utilized in the source code, and the basic functions of the blocks can be understood without extensive knowledge of graphical programming. The modular structure of the code is also visible. Figure 11 gives an overview of the flight simulation loop and initialization routines in the programming level. The flight simulation DLL is included in the sub-VI called "SIM", which is seen in the centre of the simulation loop. It should be noted that all flight mechanics calculations are performed inside the DLL. The output data is then passed to LV, which performs the routines needed for screen I/O and data storage. CPU time is calculated inside the "CPU" sub-VI. Figure 12 presents some details of the source code related to the cockpit display. The altitude indicator and artificial horizon have their own sub-VIs.

5 Conclusions

A new approach for building a graphics user interface for PC-based flight simulation software was presented in this paper. LabVIEW fulfilled the predefined requirements for the interface. The graphical programming approach is, however, not superb compared to the traditional text-based programming. Instead, it should be thought of as an alternative, especially for those who do not have experience of graphics programming. Although simple programs can be built without extensive knowledge of LV, some experience is required in order to be able to take full advantage of the graphical programming language, G. We believe, however, that the necessary expertise is much less than needed in conventional programming.

We think that the graphics capabilities of the LV were utilized quite extensively in this application. If more complex instruments are needed, probably a new approach should be chosen. Provided that the graphical presentation of the instruments shown in this paper is deemed acceptable, LabVIEW can be highly recommended.

6 Acknowledgements

The authors of this paper would like to thank the Finnish Air Force for supporting this project.

References

[1] Stevens, B. and Lewis, F., *Aircraft Control and Simulation*. John Wiley & Sons, Inc., 1992.

[2] *LabVIEW for Windows User Manual*. National Instruments Corporation, 1994.

[3] *MS Fortran Version 5.1 Advanced Topics*. Microsoft Corporation, 1991.

[4] *MS Fortran PowerStation Programmer's Guide*. Microsoft Corporation, 1995.

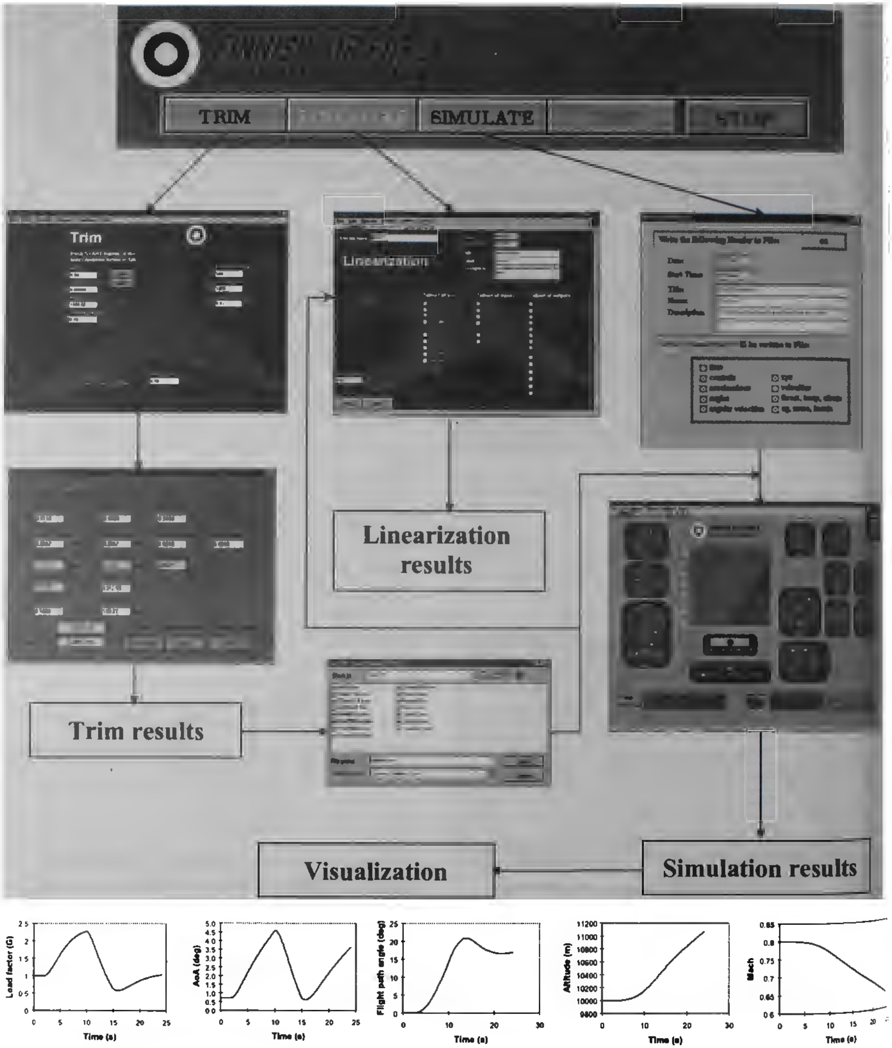


Fig. 6: An overview of the simulation session.

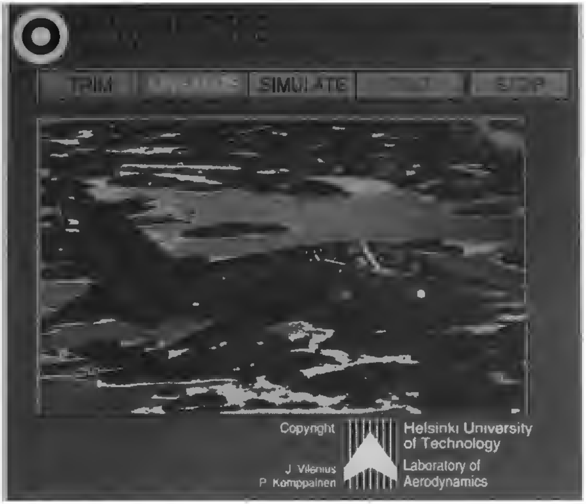
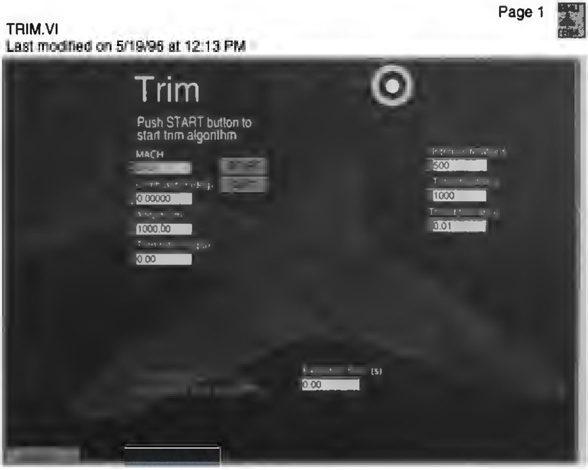


Fig. 7: The main window of HUTFLY.



Page 1

TRIM.VI
Last modified on 5/19/95 at 12:13 PM

Trim

Push START button to start trim algorithm

MACH

0.00000

1000.00

0.00

500

1000

0.01

(s)
0.00

Fig. 8: The trim window.

Page 1

Linearize.VI
Last modified on 5/19/96 at 12:30 PM

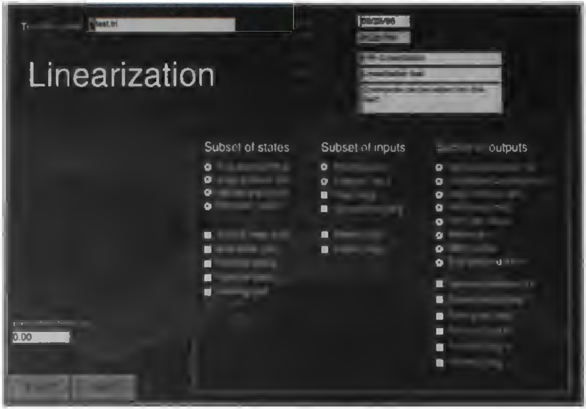


Fig. 9: The linearization window.

Page 1

SIMULATE.VI
Last modified on 5/15/96 at 2:44 PM

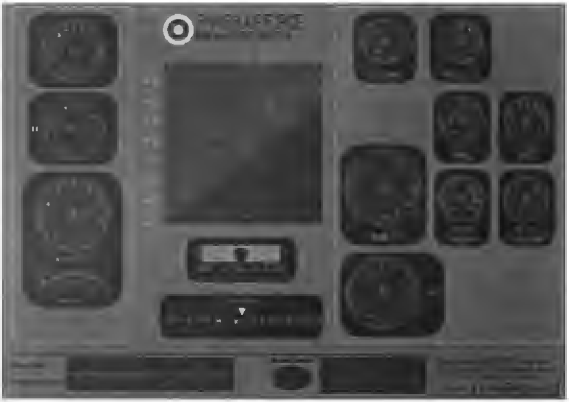
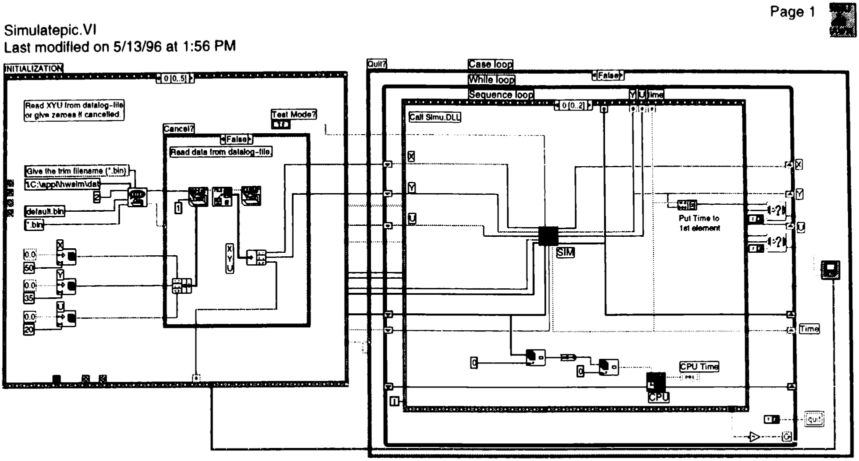
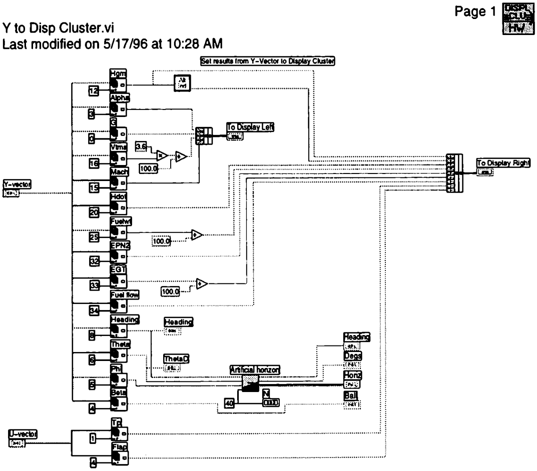


Fig. 10: The cockpit window.


Simulatepic.VI
Last modified on 5/13/96 at 1:56 PM

Page 1

Fig. 11: A detail of the flight simulation loop written in G.


Y to Disp Cluster.vi
Last modified on 5/17/96 at 10:28 AM

Page 1

Fig. 12: A detail of the cockpit display written in G.

AIAA-96-3548-CP

ADVANCED HUMAN-COMPUTER INTERFACES FOR AIR TRAFFIC MANAGEMENT AND SIMULATION

Ronald Azuma and Mike Daily
Hughes Research Laboratories
3011 Malibu Canyon Blvd. MS RL96
Malibu, CA 90265

Jimmy Krozel
Seagull Technology
21771 Stevens Creek Blvd.
Cupertino, CA 95014-1175

Abstract

New technologies will significantly change Air Traffic Control over the next 15 years. These changes will require improved human-computer interfaces for the less regulated, more complex future environment. This paper describes a highly interactive, real time demonstration of 3-D visualization and interface concepts for the air traffic domain, including Free Flight. This demonstration offers a 3-D, stereoscopic view from both the controller's and pilot's perspectives, featuring representations of projected flight paths, 3-D graphical and audio proximity warnings, and 3-D text labels that automatically reorient themselves. Feedback from domain experts is described. In Free Flight, pilots and airlines will set their own courses and resolve conflicts autonomously when possible. This demonstration also shows visualizations of the Protected Airspace Zone and Tactical Alert Zone safety regions around Free Flight aircraft, which are most easily understood through the use of 3-D graphics. Future versions of this demonstration will acquire more realistic data, improve the interaction techniques and integrate the visualization more closely with conflict detection and resolution algorithms.

Motivation

During the next 15 years, Air Traffic Control (ATC) systems will undergo significant changes due to new technologies[16]. Such changes are required to meet the expected growth in air traffic. New technologies such as the Global Positioning System (GPS), Automatic Dependent Surveillance (ADS) communications, and more sophisticated ATC software will provide some of these required improvements. These technologies will establish an "Internet in the sky," providing an information-rich environment for distributing many types of data amongst pilots and controllers. However, these technologies by themselves will not satisfy all the needs of future ATC systems. Another vital but sometimes overlooked component is the *human-computer interface*: how controllers and pilots will interact with the information provided by these new technologies. Despite increased automation and more sophisticated computers and sensors, humans will always be "in the loop." People, not computers, fly the aircraft, direct the traffic, and have the final word in all decisions. Therefore, how computer systems present information to the human controllers and pilots will play a crucial role in the effectiveness of future ATC systems.

While existing human-computer interfaces may be sufficient for today's ATC environment, they will not be in the future when air traffic patterns will be more complicated and less ordered. Existing controller interfaces use flat, 2-D displays with trackball and button-based controls. Pilots may have no traffic displays at all, relying on charts and voice commands from controllers. However, the advent of Free Flight will change the current situation dramatically. Under certain conditions, Free Flight allows pilots to set their own course and resolve conflicts by themselves, without involving controllers except when needed. This distributes the task of air traffic control over all aircraft, reducing the air traffic controllers' workloads. However, instead of the orderly well-regulated traffic patterns that exist today, there will be the potential for more complex and variable traffic patterns. Predicting and avoiding conflicts in a such an environment will be more difficult than it is today. More importantly, airlines will take an active role in air traffic management. Therefore, pilots and Airline Operation Centers (AOCs) must also have displays that show the ATC situation and potential conflicts and solutions in a way that is easy to quickly understand, without drawing concentration away from the task of flying the aircraft. Existing interfaces will not meet these needs for either pilots or controllers.

We believe the best course for meeting these future needs is to combine automated conflict detection and resolution mechanisms with advanced human-computer interfaces that use intuitive and natural three-dimensional displays and controls, such as those being developed in Virtual Reality systems. Such displays offer the potential to reduce the cognitive workload on both pilots and controllers, resulting in faster recognition and improved situational awareness of the air traffic situation and potential conflicts. These displays will be multimodal, engaging the visual, auditory and vocal channels to provide an interface that fits more naturally to the way humans normally operate, rather than forcing the human to meet the demands of the computer. The air traffic management domain provides a rich environment to study the potential of these multimodal interface techniques. This domain requires the simulation and display of a multiple aircraft airspace, intent data, proximity information for aircraft conflict detection, and conflict resolution options. In this paper, we present multimodal display techniques aiding situation understanding of the pilot and air traffic controller.

Contribution

Several investigators have applied 3-D visualization and virtual reality techniques to the air traffic management problem. Several recent studies[5, 8, 9, 11, 12] have investigated the potential benefit of 3-D or perspective displays to aid specific ATC tasks. One study[17] showed that when a display does not show the third spatial dimension as readily apparent as the other two, pilots tend to solve conflict avoidance problems in the displayed two directions more often than in the three dimensions. For conflict resolution, this implies that a plan view display may bias conflict resolution solutions to right and left turning maneuvers. A follow-on study[8] provided a perspective display to pilots for traffic avoidance and found the pilots were more likely to choose a solution with a vertical component. At least two groups have used virtual reality interfaces to examine ATC issues[3, 4, 20]. Pruyn[15] explored and demonstrated several concepts for applying 3-D visualization techniques to the ATC environment.

The contribution of this paper lies in a highly interactive, real-time demonstration of some concepts that we have not seen previously demonstrated in the ATC environment and in the lessons learned about these ideas. This is especially true of our concepts as applied to the Free Flight domain, which we have not seen discussed in any previous visualization effort. We introduced some of our concepts in a previous paper[7]; the difference here is the actual implementation, demonstration and evaluation of those concepts. Our demonstration code is based on an original version from the MIT Media Lab[18], which was greatly modified and extended by the authors to demonstrate our ideas.

Our approach differs from some previous works in that our goal is to create and demonstrate visualization concepts for the ATC domain, receive rapid feedback from domain experts, and then repeat the cycle. We have not been running user studies on incremental improvements from traditional 2-D ATC displays. We believe rapid prototyping followed by rapid evaluation is a better approach to find the more advanced visualization concepts that seem to be beneficial, which can then be formally evaluated by user studies. Furthermore, we have restricted our use of specialized equipment to the minimum required for demonstrating our concepts. While other virtual reality efforts use Head-Mounted Displays (HMDs) and fully-immersive displays, we have avoided that in favor of a stereo display on a low-end graphics workstation, even though our laboratory is equipped to run HMDs on expensive graphics engines. In this way, we have reduced the "culture shock" of exposing our ideas to controllers and pilots who are used to flat, 2-D display panels and increased the likelihood of eventual implementation of some of our concepts, due to lower cost requirements.

Initial Demonstration

The initial demonstration system is shown in Figure 1. The graphics engine is a low-cost Silicon Graphics (SGI) Indy. The user views the monitor and sees stereo images through a pair of Crystal Eyes stereo glasses, made by Stereographics. The user controls the demo through a mouse-based interface. 3-D sound is generated by two Alphatron sound boards, made by Crystal River Engineering. These boards run in a '486 PC, connected to the Indy through a serial line. The synthesized 3-D sound is routed to an audio amplifier and broadcast to a pair of 900 MHz wireless radio headphones.

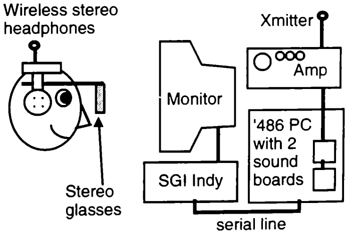


Figure 1: System diagram.

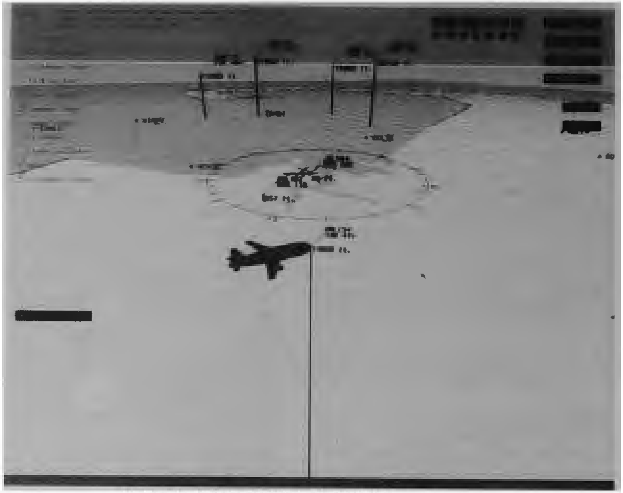


Figure 2: A sample view of what the user sees.

We first demonstrated this system on the exhibit floor of the 1995 Air Traffic Control Association Conference[1]. Figure 2 shows a typical scene from the demonstration. This system demonstrated the following features:

- *3-D Perspective Display:* Standard ATC displays show the situation in a top-down, plan-view 2-D format. This system provides that mode but also allows the user to navigate and view the scene in 3-D. The view can be exocentric, looking at the entire situation from a remote perspective, or egocentric, following an individual aircraft to see the pilot's perspective. The user controls the point of view with a mouse, using a "virtual trackball" metaphor to change the pitch and yaw of the gaze direction. Simply clicking on an aircraft or other object in the environment causes the system to focus the user's view on that object. Objects that can be clicked are indicated by pulsating 2-D icons that appear over the 3-D object when the cursor is moved over that object. Zooming the viewpoint towards or away from an object is done by pushing the appropriate buttons. The visuals can be displayed in stereo, further enhancing the 3-D effect. To provide additional depth cues, altitude lines and shadows indicate the position of each aircraft projected onto the ground. The entire database represents an area around Boston's Logan airport. The display runs at interactive rates, varying from 5-20 Hz depending on which visualization features are activated.

Certain rendering techniques are used to achieve interactive rates on a low-cost platform. Hidden-surface removal is performed primarily by using a painter's algorithm. Z-buffering is only applied to the small areas covered by the 3-D aircraft models. Textures are not supported in hardware, so grid lines are used instead to provide context for aircraft motion relative to the ground. The total polygon and line counts are kept low, and simplified models are used to render distant objects in a "level of detail" scheme.

- *Spatial and Temporal Continuity:* When the user changes viewpoints, transitions are handled smoothly. Instead of immediately changing from the current viewpoint to the selected new viewpoint, the system automatically interpolates between the old and new viewpoints. This transition takes 1-2 seconds and uses the "slow in, slow out" metaphor that slows down the rate of transition as the user leaves the initial viewpoint and arrives at the new viewpoint. Smooth viewpoint transitions aid spatial awareness by helping the user retain context with respect to the overall situation.

- *Object-Oriented Text:* In existing ATC displays, 2-D text labels are attached to aircraft and other objects.

This program also supports that mode by superimposing 2-D labels on top of their associated 3-D objects. However, another way of displaying the same information is to draw the text as 3-D objects near the labelled object. Users may perceive 3-D text labels to be more closely associated with their objects than with 2-D text labels. To be readable, these 3-D labels automatically reorient and reposition themselves based upon the user's viewpoint with respect to the object. That way, the user never has to read text that is backwards or upside down. This is done by computing the orientation of the object with respect to the current viewpoint and adjusting the label orientation and position appropriately. Figures 3 and 4 show a 3-D label (the aircraft's call sign) attached to an aircraft as viewed from two different viewpoints, demonstrating how the text automatically reorients itself to remain legible. Figure 5 shows 3-D runway labels on the model of Boston Logan.

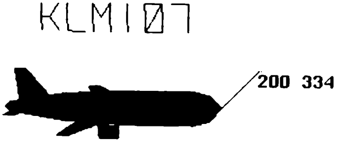


Figure 3: 3-D text label attached to aircraft.

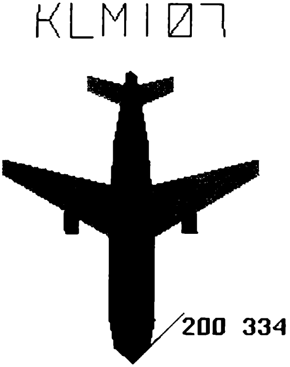


Figure 4: Reoriented 3-D text at a new viewpoint.

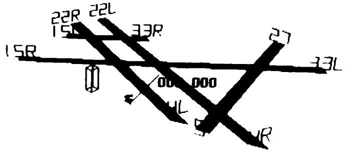


Figure 5: 3-D runway labels on Boston Logan.

One potential drawback to written information in 3-D is reduced precision in reading values along any one particular axis[19]. With object-oriented text, the orientation of the text is modified based on the viewing angle, reducing the effect of this potential problem. However, another issue that must be addressed is whether the text should remain at a fixed size or vary in size with respect to the distance from the viewpoint to the text. In this demo, the text remains at a constant size, but it could be changed to remain roughly constant in apparent screen area by scaling the 3-D font size. This prevents nearby text from occluding nearby objects and distant text from becoming illegible. Scaling the font size may also have a reinforcing effect with other motion cues.

• *Visualization of Flight Paths:* In the future, aircraft equipped with GPS antennas and ADS communications will be able to broadcast their position and intended path to control towers and other aircraft. This leads to the possibility of visualizing this information to help foresee future conflicts. This demonstration displays past and future flight paths through the use of a "ghost plane" metaphor. Ghost planes are projected future or past versions of aircraft, drawn as wireframe objects to avoid confusion with the actual aircraft (see Figure 6). Users can adjust how far into the future or past the ghost planes are projected. Since the flight paths for all aircraft are predetermined in this demo, both past and future projections use the actual paths. In reality, future projections would combine extrapolated trajectories with intent data. A wireframe line between the ghost plane and actual aircraft helps associate the two and the projected course of the aircraft (Figure 7). The ghost plane information is drawn in four different modes: static, shooting, pulsing, and airspace. Static mode simply puts the ghost plane at the projected location. Shooting mode repeatedly "shoots" the ghost plane from the actual aircraft to the projected position, following the projected path. Pulsing mode is similar, except that the ghost planes move forward and backward in time in a sinusoidal pulsing motion. By adding temporal motion, these two modes help users see how groups of aircraft will move in space and time. Finally, airspace mode represents a "highway in the sky" by drawing a set of linked wireframe rings in space that extend out as far as the projected ghost plane position (Figure 8). The result is a tunnel in the sky that the aircraft appears to fly towards (or away from), with the tunnel constantly staying ahead (or behind) the aircraft. This is useful in conflict detection by showing whether or not the airspace of two aircraft will intersect in the future, indicating a potential conflict.

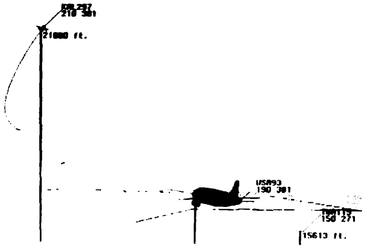


Figure 7: Several ghost plane future paths.

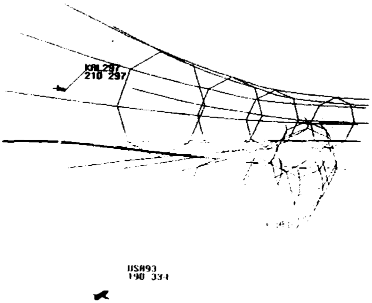


Figure 8: Ghost plane airspace mode.

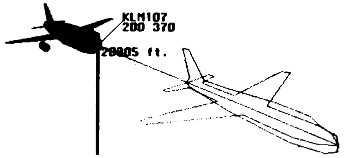


Figure 6: Ghost plane in front of actual aircraft.

- *3-D Spatial Audio Cues:* The use of 3-D sound generation hardware allows the system to attach sounds to specific objects in space. This is more than just stereo. The sounds appear to emanate from their 3-D locations, giving an additional cue for the locations of certain objects. By offering multimodal displays, users can keep track of the rough location of an object even when it leaves the visual field of view. Since we do not perform head tracking in this system, the 3-D sound computations assume the user looks straight at the workstation screen at all times. 3-D sound is used in three separate modes in this demonstration.

First, it is used as an audio alert to identify nearby aircraft. The program has a mode called "greenspace" that draws color-coded lines from the current aircraft being gazed at to other nearby aircraft: green for far away, yellow for medium distance, and red for too close. 3-D audio alerts supplement these graphic warnings, where the audio alerts are attached in space to the other nearby aircraft. 3-D audio alerts are a natural way of presenting warnings. An automobile driver is usually first aware of the presence of an ambulance by the wail of its siren and roughly estimates the location of the ambulance using sound cues. Similarly, in experiments with TCAS II collision avoidance logic, 3-D spatialized sound simulated with pilot headphones allowed pilots to acquire conflict aircraft targets approximately 2.2 seconds faster than crew members who used monotone one-earpiece headsets[2].

The second 3-D sound mode attaches an audio source to the end of one of the runways at Boston Logan. Pilots can use this as an audio beacon for keeping track of the location of a critical object in space (e.g. an airport location, a military no-fly zone, etc.) without having to look at it.

The third and last 3-D sound mode demonstrates the ability of 3-D sound to help users sort out confusing audio situations. While the user looks out at the scene from inside the control tower at Boston Logan, four conversations are played simultaneously. First, the conversations are played in mono so they all overlap each other. Listeners find it extremely difficult to focus on any of the conversations. Then we spatialize the sounds, placing each of the four conversations on one of the four main windows in the control tower (so they are separated by 90 degrees from their neighbors). As the user rotates his view within the tower, the conversations appear to rotate around his head. This takes advantage of the "cocktail party" effect: the ability to focus your hearing on a conversation that comes from a particular direction in space. Listeners find it much easier to select one conversation and focus on that with spatialized sound.

Feedback and Lessons Learned

We have not run user studies to measure the effectiveness of our ideas. Instead, we sought feedback from domain experts (controllers and pilots) who experienced the demo[1]. We learned several lessons:

- *Pilots and controllers provided different feedback:* Most controllers liked the ability to choose a static 3-D perspective view to watch the traffic from, but they did not seem to care about the interactive navigation techniques or most of the other features in the demo. A few were adamant that the 3-D displays were bad ideas and too confusing. One controller did suggest the use of 3-D sound for directing ground traffic. By making a pilot's voice appear to come from the direction of the aircraft's position on the runway, a controller issuing commands would gain an additional spatial cue to help him keep track of where the aircraft were on the runways, reducing the chance of issuing the wrong set of orders. Controllers suggested using these features in training new controllers, to help them build a "mental model" of the 3-D nature of the airspace. A few suggested it might speed up transition time when changing controllers at a station, where the incoming controller has to spend some time watching displays to understand the 3-D situation before he can relieve the controller on duty.

Pilots, on the other hand, liked most of the features, especially the visualization of future flight paths using "ghost planes" and the spatial cues for indicating neighboring aircraft. Reaction to the 3-D sound was mixed: some thought it would be useful, but others found it irritating. Military personnel thought 3-D displays could be useful for communicating information about enemy aircraft, targets on the ground, etc. that came from other sensors, such as an AWACS aircraft. Pilots suggested using this type of display in "Free Flight," when pilots will have more authority to choose their own flight path and resolve potential conflicts without the direct intervention of controllers.

These different reactions are probably due to the different requirements and biases in the two professions. Controllers train on and use 2-D displays with fixed viewpoints (no zoom capabilities) that help them quickly and consistently estimate critical distances, looking at the environment from an exocentric or "god's eye" view. This may cause a bias against 3-D displays with changing viewpoints. Pilots are used to egocentric viewpoints that see the environment from the point of view of a specific aircraft, so they were more accepting of those display modes and the attempts to aid situational awareness from the pilot's perspective.

- *Weather and terrain features:* This demo does not model weather or any terrain features. Both are factors significantly affecting air traffic and need to be considered for future visualization efforts.

- *Motion is vital for 3-D sound:* An automobile driver who hears an approaching ambulance will rotate his head to locate the siren's origin. Without motion, people have a more difficult time locating the source of a sound. Having controls that let the user rapidly change the orientation of his viewpoint was important for making 3-D sound work. Ideally, head tracking should also cue sound changes as the user turns his head.

- *Overload:* Displays and interfaces must be designed to avoid information overload. While users can turn the various display features on and off individually, it is still easy in this demo for the display to become confusing. After listening to 3-D audio alerts for a while, some users found them fatiguing and distracting. Audio alerts must be used more sparingly. Once the driver finds the ambulance, he doesn't need the siren wailing anymore. This demo has seven aircraft overall, four of which are assumed to be equipped with ADS. Even with these few, the display can become cluttered in some visualization modes. A more realistic scenario might put 40-50 aircraft in the skies around the airport. Future versions of the demo need to be more sensitive to the potential for information overload.

Free Flight Demonstration

Based on the direction of other ATC work we are performing and the feedback from the domain experts, we are focusing on building visualization aids for Free

Flight. In March 1996, the FAA decided to begin the process of shifting the current air traffic control system to a new policy of air traffic management called Free Flight[13]. Free Flight shifts the emphasis from active to passive control, with a policy of intervention by exception[14]. Under certain circumstances, aircraft will be allowed to select their own routes to their destinations, provided that they do not conflict with other aircraft. This freedom to set courses could result in significant time and financial savings for commercial airlines. When potential conflicts occur, Free Flight policies emphasize having aircraft attempt to resolve conflicts by themselves, with air traffic managers (who oversee dynamically-defined airspace sectors) intervening only when required.

What constitutes a conflict? To ensure safety, no aircraft should penetrate another aircraft's Protected Airspace Zone (PAZ): a cylindrical region around each aircraft with a 5 nautical mile radius and total height of 2000 feet (Figure 9). To avoid penetrating a PAZ, aircraft can change velocity, heading, or altitude. The Tactical Alert Zone (TAZ) is a region around an aircraft that defines the space where it is no longer possible to execute a maneuver to avoid penetrating the PAZ. For example, see Figure 10. Aircraft B is inside aircraft A's Velocity TAZ. That means even if both aircraft A and B cooperate and optimally change their velocities, penetrating the PAZ is inevitable due to limitations on aircraft performance. However, aircraft B is still outside aircraft A's Heading TAZ, so it is possible to change headings and avoid penetrating the PAZ. In the worst case, if it is impossible to avoid penetrating the PAZ, then any available control (velocity, heading, altitude) must be used to maximize the miss distance.

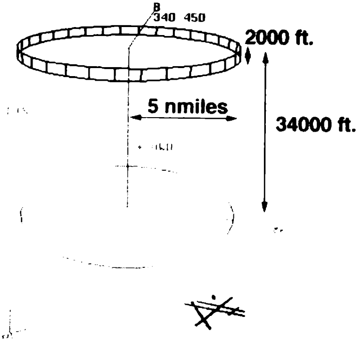


Figure 9: Protected Airspace Zone (PAZ).

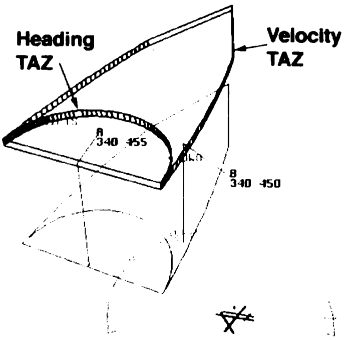


Figure 10: Tactical Alert Zones (TAZs).

The TAZ are complex, constantly changing shapes that are not easily explained or communicated without the use of 3-D graphics. Computing the shape of the TAZ requires knowledge of performance limits (maximum turn rate, maximum climb rate, etc.) and controllability and reachability constraints derived from the relative equations of motion. These computations are described in another paper[10]. The PAZ and TAZ are important safety regions that both pilots and air traffic managers must clearly understand to perform conflict detection and resolution in Free Flight. Both the PAZ and TAZ are centered on a particular aircraft and move with the aircraft. While the PAZ does not change shape with time, the TAZ will change depending on the relative positions, headings, speeds and conditions of the two aircraft. The TAZ cannot be easily generalized in terms of a heuristic that a pilot or air traffic manager can use in a non-visual mode. The best way to make use of the theoretical and algorithmic backgrounds behind computing the TAZ is to display the TAZ regions around the aircraft in a visual manner.

To illustrate TAZ regions for heading control manuevering, we have created a three-aircraft tactical conflict detection and resolution example. Three aircraft, labelled A, B, and C, are headed on courses that will cause them to penetrate each other's PAZ regions. These aircraft are too close to initiate speed control manuevers and are currently being told to stay within the same flight level by air traffic management. Figures 11-13 show what happens if the aircraft do not execute any heading changes. Figures 14-19 show how things change if the aircraft cooperate and adjust their headings to avoid penetrating each other's safety zones. In this case, the conflict resolution maneuver performed by aircraft A and B leads to a secondary conflict and subsequent resolution between A and C, allowing for the safe passage of all three aircraft. In all these figures we see the TAZ regions around aircraft B or C with respect to possible penetration by aircraft A. Ghost planes extended 30 seconds into the future show the intended paths of each aircraft. By displaying what will happen with no course change, these visual displays provide advance warning of potential conflicts and allow the pilots and the air traffic managers to understand the control advice provided by the TAZ regions. This three-aircraft scenario is a particularly difficult multi-aircraft conflict that might occur under Free Flight. Investigating such scenarios provides a way to research multiple aircraft conflict detection and resolution algorithms and visualization techniques to aid situational awareness and the presentation of conflict resolution advisory information.

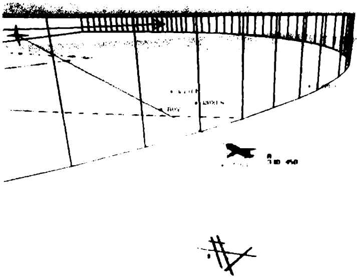


Figure 12: 3 aircraft scenario, no maneuvering. Time = 38 seconds. Pilot A's view of impending contact with aircraft B's Heading TAZ.

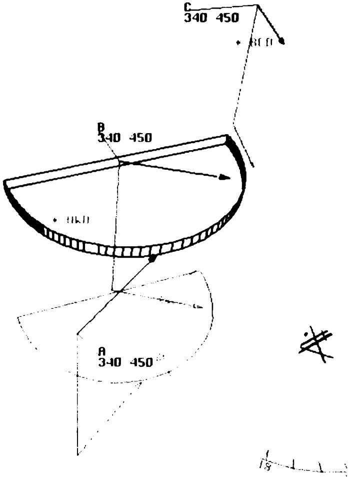


Figure 11: 3 aircraft scenario, no maneuvering. Time = 10 seconds.

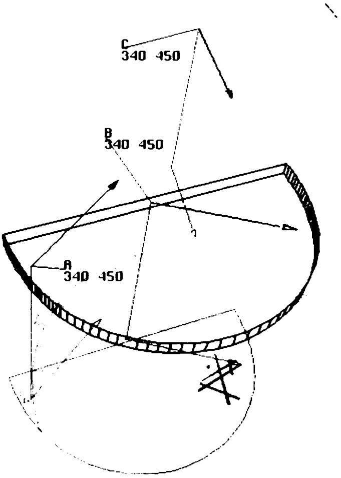


Figure 13: 3 aircraft scenario, no maneuvering. Time = 55 seconds. Aircraft A has penetrated aircraft B's Heading TAZ.

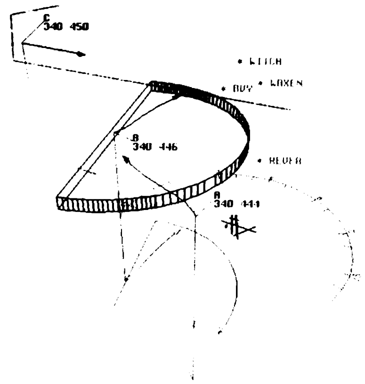


Figure 14: 3 aircraft scenario, with maneuvering. Time = 33 seconds. Aircraft A and B both start turning to avoid penetration of B's TAZ.

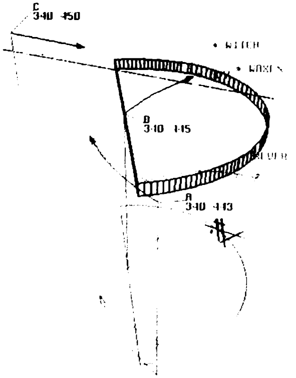


Figure 15: 3 aircraft scenario, with maneuvering. Time = 46 seconds. Aircraft A is almost finished clearing aircraft B's TAZ.

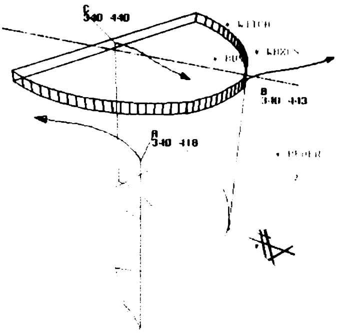


Figure 16: 3 aircraft scenario, with maneuvering. Time = 80 seconds. Aircraft A has cleared aircraft B but is now approaching aircraft C. Aircraft A and C begin cooperative turn to avoid penetration of C's TAZ.

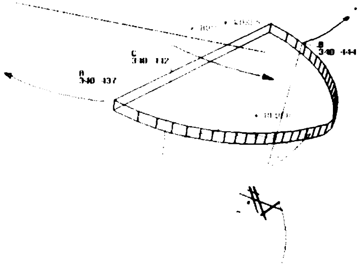


Figure 17: 3 aircraft scenario, with maneuvering. Time = 103 seconds. Aircraft A has just finished clearing aircraft C's TAZ.

Notes on Figures 11-18: All three aircraft stay at 34000 feet and are labelled A, B, and C on the diagrams. Altitude lines extend from the three aircraft positions to the ground. Shadows for the TAZ regions and the "ghost plane" lines are drawn on the ground. The lines extending from the aircraft positions and terminating in an arrow are the ghost plane lines. These visualizations are easier to see on the workstation monitor because the user can interactively change his viewpoint in real time and can see them as 3-D stereo images.

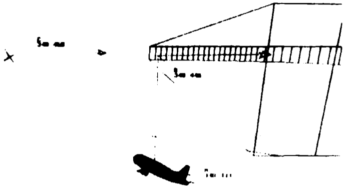


Figure 18: 3 aircraft scenario, with maneuvering. Time = 53 seconds. Pilot A's view just after clearing aircraft B's TAZ.

Future Work

We have the following plans to continue developing visualization and interface techniques for aiding Free Flight:

- *New hardware and software infrastructure:* The existing demonstration is written in the C language using the GL graphics library. Consequently, adding new features requires changing low-level code, which is time consuming. We will switch to a higher-level software toolkit that is built for 3-D display and interaction, such as Sense 8's WorldToolKit or Division's dVISE library. This should result in faster changes and more rapid prototyping. We also anticipate moving to a more capable hardware platform, such as an SGI Infinite Reality, to avoid restricting our visualization ideas to the performance limitations of our SGI Indy. We will also build a version of the code that shows images on a wide field-of-view immersive stereo display. Hughes Research Labs has a 160 degree field-of-view toroidal screen driven by three projectors. While not appropriate for pilots, the air traffic manager or AOC personnel of the future might stand in front of such a display and take advantage of the large visual display area provided. Such displays are sometimes referred to as CAVEs[6].

- *Improved interaction:* The existing mouse-based interface may be satisfactory for a workstation-based demo, but as we move to large-screen displays this interface will no longer be appropriate. Selecting objects and changing modes should be allowed by issuing voice commands, rather than clicking objects on the display. Speech recognition has progressed to the point where small vocabulary speaker-independent recognizers, suitable for this task, are available on PCs. Head tracking will be required to correctly render the scene and allow user gaze to select objects. We will also track hand locations or handheld objects that the user manipulates to allow gesture-based control mechanisms.

- *More realistic data:* The current demonstration has only seven aircraft, flying routes that were not modelled according to existing flight paths or physical simulation. We will acquire more realistic databases of an airspace with flight routes from real commercial aircraft. We must populate the airspace with more aircraft, around 40 or 50, to more realistically simulate the congestion that can exist around busy airports.

- *Integration with conflict detection and resolution:* Airspace congestion can lead to operator overload problems. To avoid this overload, filters will be used so that the user will only see the information that is critical. A pilot in Free Flight does not need to keep track of all neighboring aircraft, but rather only the ones that threaten to result in a conflict. Conflict detection and resolution algorithms can provide that information while improved visualization reduces operator overload.

- *More visualization modes:* More work needs to be done in exploring additional ways of representing the TAZ regions. What are the best ways to show the TAZ for heading, velocity, and altitude changes? The visualizations might be different for each type of zone.

References

[1] 40th Annual Air Traffic Control Association Conference. (Las Vegas, NV, 10-14 September 1995).

[2] Begault, D. R. Head-Up Auditory Displays for Traffic Collision Avoidance System Advisories: A Preliminary Investigation. *Human Factors 35*, 4 (December 1993), 707-717.

[3] Brown, Mark A. On the Evaluation of 3D Display Technologies for Air Traffic Control. Queen Mary & Westfield College Dept. of Computer Science Technical Report No. 682 (16 August 1994).

[4] Brown, Mark A. 3D Display Technologies for Air Traffic Control: A Pilot Study. Queen Mary & Westfield College Dept. of Computer Science Technical Report No. 690 (12 December 1994).

[5] Burnett, Meridyth Svensa and Woodrow Barfield. Perspective versus plan view air traffic control displays: Survey and empirical results.

Proceedings of Human Factors Society 35th Annual Meeting (1991), 87-91.

[6] Cruz-Neira, Carolina, Daniel J. Sandin, and Thomas A. DeFanti. Surround-Screen Projection-Based Virtual Reality: The Design and Implementation of the CAVE. *Proceedings of SIGGRAPH '93* (Anaheim, CA, 1-6 August 1993), 135-142.

[7] Daily, Michael J. and Jimmy A. Krozel. Human-Centered Interfaces for Air Traffic Control. *Proceedings of the 40th Annual Air Traffic Control Assoication (ATCA) Conference* (Las Vegas, NV, 10-14 September 1995), 215-219.

[8] Ellis, Stephen R. and Michael W. McGreevy. Perspective Traffic Display Format and Airline Pilot Traffic Avoidance. *Human Factors 29*, 2 (August 1987), 371-382.

[9] Haskell, I.D. and Christopher D. Wickens. Two- and Three-Dimensional Displays for Aviation: A Theoretical and Empirical Comparison. *International Journal of Aviation Psychology 3*, 2 (1993), 87-109.

[10] Krozel, Jimmy, T. Mueller and G. Hunter. Free Flight Conflict Detection and Resolution Analysis. To be published in *Proceedings of AIAA Guidance, Navigation and Control conference* (San Diego, CA, 29-31 July 1996).

[11] May, Patricia A., Margaret Campbell and Christopher D. Wickens. Perspective Displays for Air Traffic Control: Display of Terrain and Weather. *Air Traffic Control Quarterly 3*, 1 (1995), 1-17.

[12] Moller, H. and G. Sachs. Synthetic Vision for Enhancing Poor Visibility Flight Operations. *IEEE AES Systems Magazine* (March 1994), 27-33.

[13] Nomani, Asra Q. FAA to Let Pilots Change Flight Paths. *The Wall Street Journal* (15 March 1996), A3.

[14] Nordwall, Bruce D. Free Flight: ATC Model for the Next 50 Years. *Aviation Week & Space Technology* (31 July 1995), 38-39.

[15] Pruyn, Peter W. and Donald P. Greenberg. Exploring 3D Computer Graphics in Cockpit Avionics. *IEEE Computer Graphics and Applications* (May 1993), 28-35.

[16] Scardina, John A., Theodore R. Simpson, and Michael J. Ball. ATM: The only constant is change. *Aerospace America* (March 1996), 20-23 and 40.

[17] Smith, J.D., Stephen R. Ellis and Edward Lee. Avoidance Manuevers Selected While Viewing Cockpit Traffic Displays. NASA Ames Research Center technical report TM-84269 (October 1982), 34 pages.

[18] Ventrella, Jeffrey and Robert McNeil. MIT Media Laboratory, personal communication (1994).

[19] Wickens, C. D., S. Todd and K. Seidler. Three-Dimensional Displays: Perception, Implementation, and Applications. University of Illinois at Urbana-Champaign Aviation Research Laboratory technical report ARL-89-11 (1989).

[20] Zeltzer, David and Steven Drucker. A Virtual Environment System for Mission Planning. *Proceedings of the IMAGE VI conference* (Scottsdale, AZ, 14-17 July 1992), 125-134.

A NUMERICAL STUDY OF TURBULENCE EVENTS ABOVE THE ATMOSPHERIC BOUNDARY LAYER

Darko Koračin, Nash'at Ahmad, Vlad Isakov, *Desert Research Institute, Reno, NV*
Luis Mendez-Nuñez, *University of California, Davis, CA*

Abstract

A 2-D fully compressible atmospheric model has been developed to study dynamical and thermal instabilities within the planetary boundary layer and free atmosphere. We present a preliminary model evaluation and the results from the numerical experiment designed for a study of the effects of aircraft exhaust plumes on generation of local dynamical and thermal instabilities. The results of sensitivity tests with and without treatment of the turbulent diffusion in the model equations are also reported. We found that the dynamical instabilities are more dominant than the thermal instabilities. The simulation results also showed the sinusoidal-type of instabilities and vortex formation in the aircraft exhaust plume as evidenced by observational studies.

1 Introduction

Turbulence events above the planetary boundary layer are intermittent and consequently difficult to observe and simulate. The presence of aircraft and spacecraft imposes strong modification to local atmospheric stability and dynamics. The modification of local dynamics, in particular vertical motions, determines eventual cloud formation and evolution. One of the significant examples is prediction of contrails, which are important phenomena to aviation operations. Recently, we have developed a fully compressible atmospheric model that can be used in numerical studies of both the planetary boundary layer and the free atmosphere. The model results from a preliminary study of effects of broken clouds on radiative heat transfer within the atmospheric boundary layer are reported by Koračin et al. (1996)[1] and Isakov et al. (1996)[2]. The main objectives of this study are: to evaluate the model; to simulate generation of local vertical velocity and waves due to aircraft flight patterns above the atmospheric boundary layer; and to estimate the importance of turbulent diffusion on perturbation of dynamical fields. The model will enable us to study the effects of high speed objects on atmospheric stability and critical conditions for formation of ice-containing clouds.

2 The Model

2.1 Basic Equations

The model is fully compressible and based on the set of 2D equations describing conservation of mass (the continuity equation), conservation of momentum (equations of motion), conservation of heat (the thermodynamic equation), and the ideal gas law:

$$\frac{\partial \rho}{\partial t} + \frac{\partial \rho u}{\partial x} + \frac{\partial \rho w}{\partial z} = 0$$

$$\frac{\partial \rho u}{\partial t} + \frac{\partial u \rho u}{\partial x} + \frac{\partial w \rho u}{\partial z} = -\frac{\partial P}{\partial x} + T_x$$

$$\frac{\partial \rho w}{\partial t} + \frac{\partial u \rho w}{\partial x} + \frac{\partial w \rho w}{\partial z} = -\frac{\partial P}{\partial z} + T_z - \rho g$$

$$\frac{\partial \rho \Theta}{\partial t} + \frac{\partial u \rho \Theta}{\partial x} + \frac{\partial w \rho \Theta}{\partial z} = T_\theta$$

$$P = P_0 \left(\frac{R_d \rho \Theta}{P_0} \right)^{\frac{C_p}{C_v}}$$

where u and w are wind components in X and Z directions, respectively; ρ is air density, T_x, T_z, and T_θ are the terms of turbulent diffusion; g is the acceleration of gravity; Θ is the potential temperature; T is temperature; C_p and C_v are the specific heat at constant pressure and volume, respectively; P is the atmospheric pressure; and R_d is the gas constant for dry air.

2.2 Numerical Solution

The solution for air density, wind components, and potential temperature is obtained using a MacCormack finite difference scheme (MacCormack 1969)[3] for a non-linear system of equations listed in the previous sub-section. Each time step consists of predictor and corrector sub-steps. According to Mendez and Carroll (1993, 1994)[45], this scheme compares well with the Smolarkiewicz and leap-frog schemes.The MacCormack scheme is of a second-order accuracy and does not require filtering or grid staggering. Boundary conditions are open, and the selected grid resolution determines the time step that is necessary for the convergence of the solution according to the Courant number.

The nonlinear system of equations can be written in a general form:

$$\frac{\partial \mathbf{V}}{\partial t} + \frac{\partial \mathbf{E}}{\partial x} + \frac{\partial \mathbf{F}}{\partial z} = \mathbf{H} + \mathbf{T}$$

where

$$\begin{aligned}
\mathbf{V} &= (\rho, \rho u, \rho w, \rho \theta) \\
\mathbf{E} &= (u\rho, u\rho u + P, u\rho w, u\rho\theta) \\
\mathbf{F} &= (w\rho, w\rho u, w\rho w + P, w\rho\theta) \\
\mathbf{H} &= (0, 0, -\rho g, 0) \\
\mathbf{T} &= (0, T_u, T_w, T_\theta)
\end{aligned}$$

where T_u, T_w, and T_θ are sub-grid contributions to the the grid-resolving solution of the model equations. These turbulence diffusion terms are explained in the next section.

The predictor sub-step is:

$$\begin{aligned}
V^*_{i,k} = V^n_{i,k} &- \frac{\Delta t}{\Delta x}(E^n_{i+1,k} - E^n_{i,k}) \\
&- \frac{\Delta t}{\Delta z}(F^n_{i,k} - F^n_{i,k-1}) \\
&+ \frac{\Delta t}{2}(H^n_{i,k} + H^n_{i,k-1}).
\end{aligned}$$

The corrector sub-step is:

$$\begin{aligned}
V^{n+1}_{i,k} &= \frac{1}{2}[V^n_{i,k} + V^*_{i,k} \\
&- \frac{\Delta t}{\Delta x}(E^*_{i,k} - E^*_{i-1,k}) \\
&- \frac{\Delta t}{\Delta z}(F^*_{i,k+1} - F^*_{i,k}) \\
&+ \frac{\Delta t}{2}(H^*_{i,k+1} + H^*_{i,k})].
\end{aligned}$$

where i and k are the indices in the X and Z directions, respectively. To eliminate the bias due to this one-sided differencing, the forward and backward differencing is sequentially alternated between predictor and corrector steps at each succesive time step. This procedure preserves the symmetry of the solution; it is important since no filtering is imposed in the model.

We tested a wide range of resolution, from 10 meters to several kilometers in both the horizontal and vertical dimensions. The model is being run on both a PC and a work station.

2.3 Turbulence Closure

In order to determine the effect of turbulent diffusion on the subgrid scale, turbulent fluxes are parameterized according to Smagorinsky (1963)[6]. The essence of the method is to calculate deformation rate and relate it to the eddy viscosity and turbulent fluxes. The turbulence term $\mathbf{T}$ is:

$$T_j = \frac{\partial \tau_{ij}}{\partial x_i}$$

where j is the coordinate index and τ_{ij} are the components of the Reynolds stress tensor.

The components of the Reynolds stress tensor can be expressed as:

$$\tau_{ij} = \rho K_m D_{ij}$$

The components of the deformation tensor D_{ij} are calculated as:

$$D_{ij} = \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} - 2\frac{\delta_{ij}}{\delta_{ii}}\frac{\partial u_k}{\partial x_k}$$

Eddy viscosity coefficient, K_m, is related to the total deformation by the following:

$$K_m = \begin{cases} \frac{(c\Delta)^2}{\sqrt{2}}|D_t|(1 - Ri)^{1/2} & \text{if } Ri < 0 \\ 0 & \text{otherwise.} \end{cases}$$

Total deformation rate D_t is defined as:

$$D_t^2 = \frac{1}{2}\sum_i \sum_j D_{ij}^2$$

The Richardson number is defined as:

$$Ri = \frac{g}{\Theta}\frac{\frac{\partial\Theta}{\partial z}}{\left[\left(\frac{\partial u}{\partial z}\right)^2 + \left(\frac{\partial v}{\partial z}\right)^2\right]}$$

δ_{ij} is the Kronecker symbol:

$$\delta_{ij} = 0 \text{ for } i \neq j$$

$$\delta_{ij} = 1 \text{ for } i = j$$

Θ is the potential temperature, c is the constant (0.28) and Δ is the horizontal resolution (m) of the model grid.

Both the mechanical and thermal diffusion terms are calculated using second order centered difference.

3 Model Evaluation

We compared the model results with the results from the compressible model developed by Droegemeier (1985)[7]. The comparison was performed by using an idealized experiment of buoyancy due to a "hot" bubble in an isotropic environment. The model grid consisted of 80 horizontal and 60 vertical points with both a horizontal and vertical resolution of 500 meters. The time step was 0.5 seconds. A "hot" bubble with a radius of 2500 m and the maximum temperature excess of 6.6 K in the bubble center was initialized close to the surface. The temperature excess within the bubble decreased linearly toward the edge to where it matches an isotropic environment (300 K). This positive thermal perturbation develops strong pressure gradients, horizontal and vertical wind fields in a neutral environment. Consequently the bubble is vertically displaced by buoyancy and disintegrates in time. Fig. 1 shows contour plots of initial thermal perturbation (spherical bubble), u and w wind components, and pressure perturbation fields. The same case study was simulated by Droegemeier (1985)[7] and his results are shown in Figs. 4.1a-c (p. 167-173). It can be seen that the models agree quite well, in spite of the fact that our model does not require any explicit filtering and has fully open lateral and top boundaries. We also performed the model comparison on a case study of the same bubble in a stable environment. Figures are not shown here and will be reported elsewhere. The models also agree fairly well in that case.

4 Numerical Experiment

The main objective was to design the numerical experiment to study effects of aircraft flight pattern on generation of local vertical velocity and waves above the atmospheric boundary layer. The model grid consisted of 300 horizontal and 200 vertical points. The resolution in both horizontal and vertical directions was 10 m. Time step of 0.005s was used. The bottom boundary of the numerical model was taken at 500 hPa. The environmental potential temperature lapse rate was $3\text{K} \cdot \text{km}^{-1}$. Characteristic properties of a jet exhaust plume in the atmosphere are still not well known. Pitchford et al. (1991)[8] presented the results from the ***in situ*** measurements of the atmospheric parameters and aerosols in the jet plume. They measured the excess in environmental temperature due to jet plume of about 2 – 4 degrees at $2600mAGL$. This result was obtained at distance of about $200m$ behind the aircraft. Since our model was run with both the horizontal and vertical resolution of $10m$, we decided to simulate larger temperature difference of $10K$ between the exhaust and environment in the first few points at the aircraft location. In addition, we are considering higher altitudes where the larger differences could be expected. The thermal and dynamical perturbation started on the west boundary at $z = 1000$ m. This perturbation ("hot point"/"aircraft") was horizontally translated with a speed of 500 $m \cdot s^{-1}$. At 6 seconds it reached the eastern boundary, leaving the numerical domain. The evolution of the perturbed environment after this time was studied.

An aircraft propagating at such speed in an environment at rest produces rapidly propagating pressure waves (shock waves); a series of maxima and minima can be observed. Figure 2 shows the pressure perturbations with regard to the initial hydostatic environment at 1.75 and 5.75 seconds. It can be seen that the pressure wave at the top of the domain goes right through the open upper boundary. No sponge layer was used in the model. On the other hand, the shock wave is reflected by the closed lower boundary. The reflected wave leaves the model domain through the open upper and right boundaries at about 14 seconds (not shown).

Figs. 3 and 4 show the time evolution of the predicted u- and w-wind fields. At 12 seconds (top panels), it has been 6 seconds since the aircraft left the right boundary of the domain. In the region close to this boundary, the wake widened from 20 to about 120 meters in the vertical direction, without generating significant instabilities. Diffusion processes resulting in important up- and down-drafts at 1000 meters east of the western boundary (10 seconds after the aircraft passage). At that location, the wake had been vertically displaced. Turbulent processes of the wake become even more apparent at later times.

Fig. 5 shows the simulated u-wind field without the parameterized turbulence diffusion. The lack of vertical diffusion resulted in a much lower vertical spreading of the wake. At 14 seconds some vortices above and below the wake can be seen; the generation of these vortices cannot be associated to turbulent processes, but rather to the strong mechanical shear and buoyant effects between the wake and the environment. At 18 seconds (lowest panel), the vortices above and below the wake are strengthened but the wake is still constrained to just a few grid points. Comparing Fig. 5 and Fig. 3, it can be seen that the treatment of turbulence diffusion is essential in the studies of aircraft jet plumes. The behavior of the simulated wind fields in the aircraft plume agrees well with the sinusoidal-type of dynamical instabilities as suggested by Crow (1970)[9] and others.

5 Conclusions

A 2-D fully compressible model has been applied to the evolution of jet exhaust plumes and associated dynamical and thermal instability. The present model results agree well with the results from Droegemeier's numerical experiment of the buoyant bubble. The model was able to reproduce atmospheric structure in the case of large speeds of the simulated aircraft and its highly buoyant plume. The preliminary results indicate a possible mechanism for initiation of local vertical velocity and waves. Buoyant effects and wind-shear gradients with a time delay due to plume propagation are causing waves and eventual formation of vortices. Dynamical instabilities are initiated by the strong local vertical shear in the u-wind component. A compensating horizontal shear is then created, and it initiates the horizontal inhomogeneities and instabilities in the plume. Buoyancy is then superimposed on this effect, but according to our preliminary simulations, it might be of secondary importance compared to dynamical instabilities. This also confirms the findings by Crow (1970)[9], who studied the effect of dynamical instabilities and vortex generation in the trailing aircraft plume. According to our simulations, turbulent diffusion is a significant component in redistributing dynamical and thermal instabilities and the consequent evolution of the jet exhaust plume.

References

[1] Koračin, D., Isakov, V., and Mendez-Nuñez, L., 1996: "A cloud-resolving model with the radiation scheme based on the Monte Carlo method." To be presented at the *12th International Conference on Clouds and Precipitation,* 19-23 August 1996, Zürich, Switzerland.

[2] Isakov, V., Koračin, D., and Mendez-Nuñez, L., 1996: "A radiation scheme based on the Monte Carlo method for a new cloud-resolving model." To be presented at the *International Radiation Symposium,* 19-24 August 1996, Fairbanks, Alaska.

[3] MacCormack, R.W., 1969: "The effect of viscosity in hypervelocity impact cratering." textitA-IAA Paper No. 69-354.

[4] Mendez-Nuñez, L. R., and Carroll, J. J., 1993: "Comparison of Leapfrog, Smolarkiewicz, and MacCormack Schemes Applied to Nonlinear Equations." *Mon. Wea. Rev.*, **121**(2): 565–578.

[5] Mendez-Nuñez, L. R., and Carroll, J. J., 1994: "Application of the MacCormack Scheme to Atmospheric Nonhydrostatic Models." *Mon. Wea. Rev.*, **122**(5): 984–1000.

[6] Smagorinsky,J., 1963: "General circulation experiments with the primitive equations." *Mon. Wea. Rev.*, **91**(5): 99–164.

[7] Droegemeier, K. K., 1985: "The numerical simulation of thunderstorm outflow dynamics." Ph.D. dissertation, University of Illinois at Urbana-Champaign, 695 pp.

[8] Pitchford, M., Hudson, J.G., and Hallett, J., 1991: "Size and critical supersaturation for condensation of jet engine exhaust particles." *J. Geoph. Res.*, **96**, 20,787-20,793.

[9] Crow, S.C., 1970: "Stability theory for a pair of trailing vortices." *AIAA Journ.*, **8**, 2172-2179.

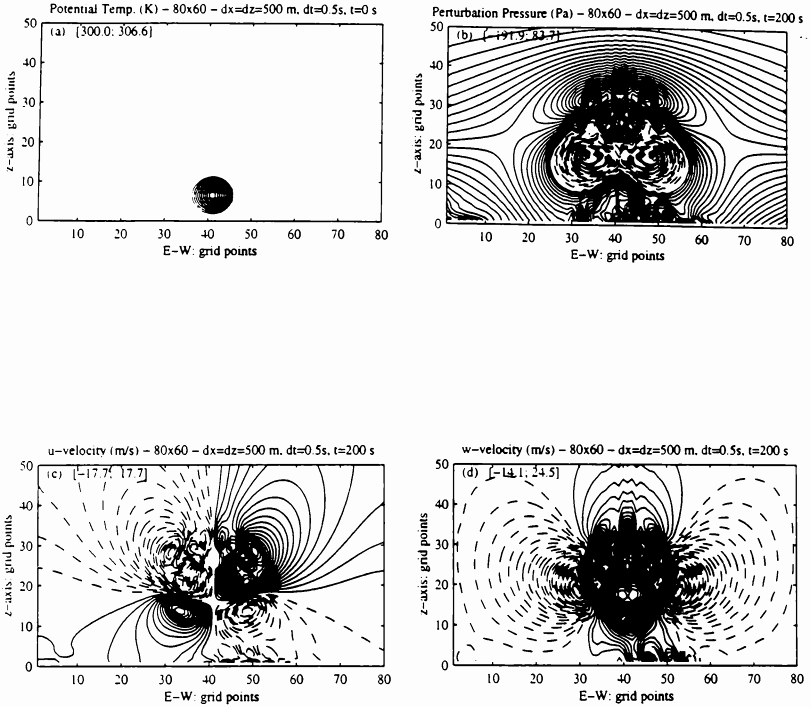


Fig 1: Contour plots of initial potential temperature (K) field (a), perturbation pressure (Pa) 200 s after the simulation start (b), u-component of wind (m/s) 200 s after the simulation start (c) and w-component of wind (m/s) 200 s after the simulation start (d). Dashed lines represent negative values and solid lines represent positive values. Minimum and maximum data values of each plot are indicated in the top-left corner.

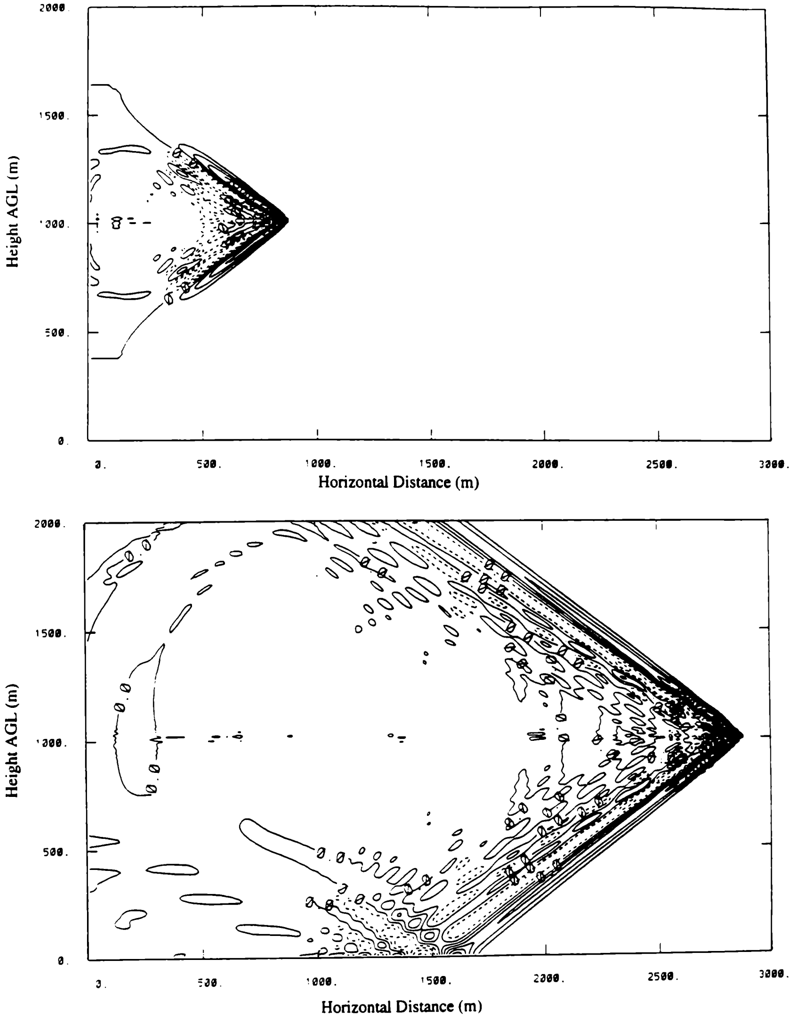


Fig 2. Contour plots of pressure perturbation (Pa) at 1.75 s (top panel) and 5.75 s (bottom panel).

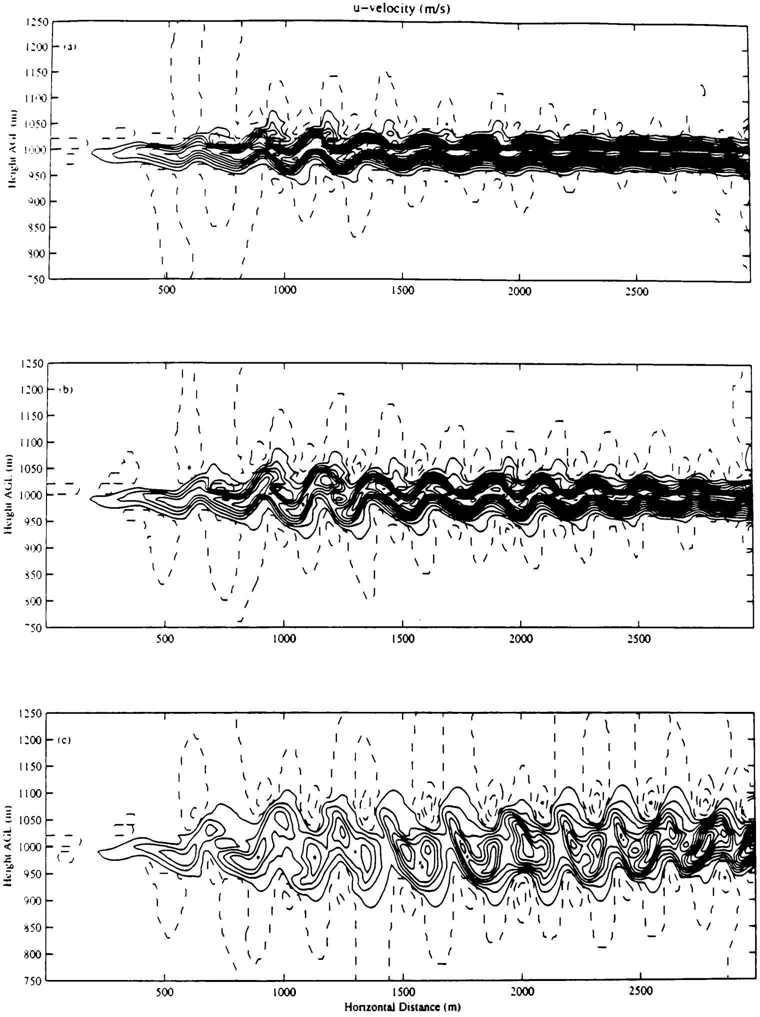


Fig 3. a, b & c: Contour plots of simulated horizontal velocity (m/s) at 12, 14 and 18 seconds respectively. Only the 50 vertical levels, immediately adjacent to the aircraft trajectory are shown. The contour plots range is from -15 m/s to 70 m/s with increments of 5 m/s.

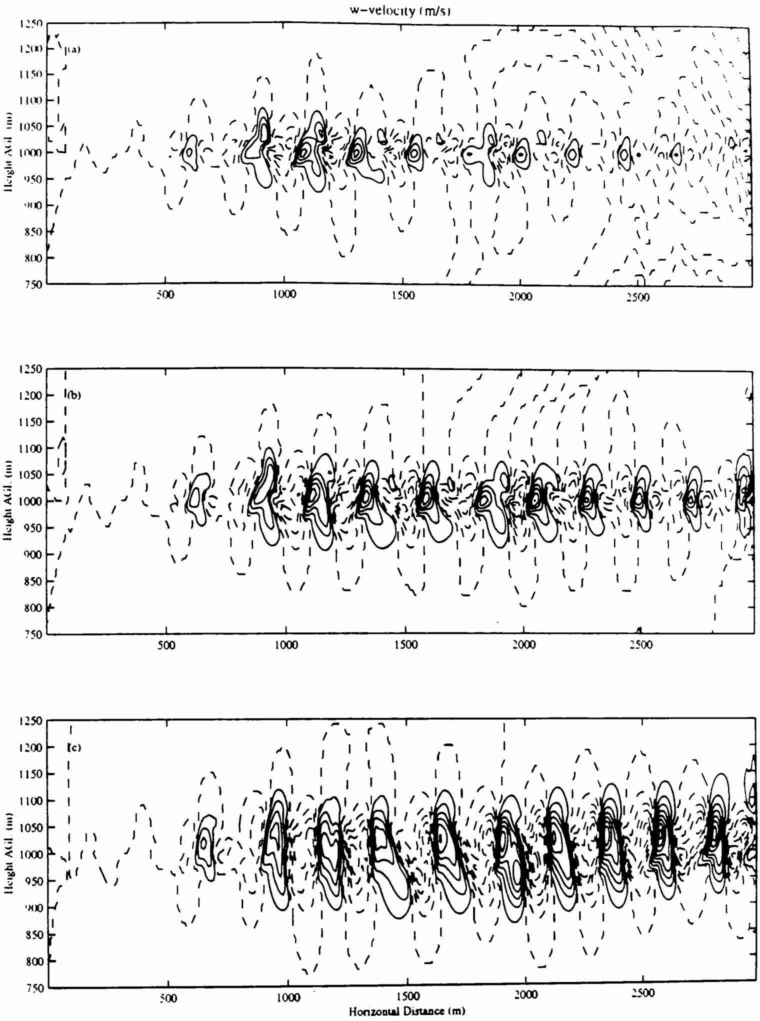


Fig 4. a, b & c: Same as Fig 3, but for the vertical velocity (m/s). The contour plots range is from -30 m/s to 60 m/s with increments of 5 m/s.

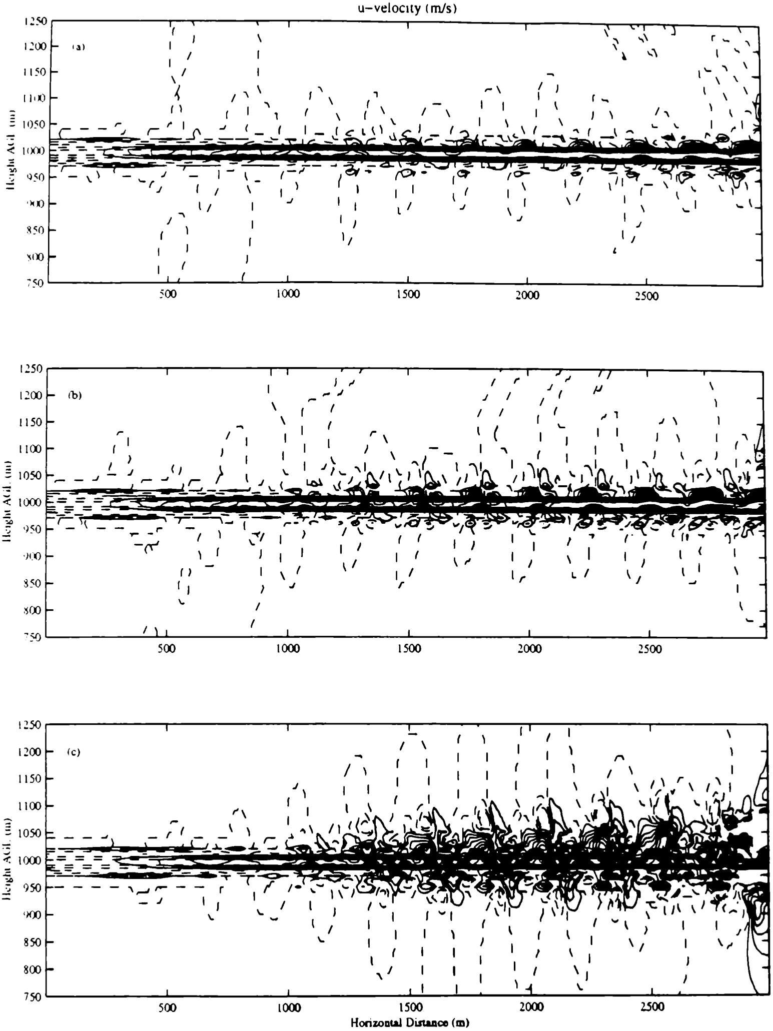


Fig 5. a, b & c: Same as Fig 3, but in this case no turbulent diffusion parameterization was used. The contour plots range is from -15 m/s to 70 m/s with increments of 5 m/s.

AIAA-96-3550-CP

HIGH VISUALIZATION CAPACITIES FLIGHT MINISIMULATOR WITH DEMONSTRATED PROTOTYPE OF OPERATOR MOTOR ACTION SUPPORT SYSTEM

Dr. Oleg A.Yakimenko, Vladimir V.Demidov
Military-Air Engineering Academy n.a. Prof. N.E.Zhukovskiy, Moscow, Russia

Artur A.Dyshalenkov, Alexander P.Nikitin, Semen L.Semenov
Joint-stock "Tekhno", Moscow, Russia

It's considered in the paper the main features of realization and functioning of low cost high outside cabin environment visualization capacities minisimulator with prototype of pilot's motor actions support system based on powerful Pentium-processor personal computer (it was shown recently on Dubai-95 exhibition and received high pilot's marks). It's discussed the apparatus realization and illustrated capacities of ground surface (outside cabin environments) visualization, put forward main ideas of designed demonstration prototype of pilot's support system during maneuvering (approaches for typical maneuvers tasks formalization and "trajectories banks" formation; recommended trajectory for current conditions real time reconfiguration; "road in the sky" generation and utilization - following in "director with sight" mode - for piloting skills obtaining).

Introduction

The role of ground simulators in the modern processes of design, flight testing and operational development of complex objects of aviation technique is a matter of common knowledge. The more importance significance simulators have in the procedure of pilots learning - from primary education up to facility improvement.

Last achievements in computers design open up new possibilities for chip and simple in use minisimulators development, which can solve a great number of tasks being earlier tackled by complex installations.

Present article makes up the common description of flight minisimulator designed by joint efforts of specialists of Zhukovskiy Military-Air Engineering Academy and joint-stock "Tekhno", which is used for mentioned purposes.

General characteristics

General structure of designed minisimulator is shown on Fig.1.

Its hardware includes Pentium/133-processor with 16MB RAM and 1MB SVGA videocard (256 colors palette). Operator control aircraft model movement through standard peripheral analog apparatus F-16, WCS and Rudder of Thrust Master Corp.; and through discrete commands from keyboard.

Software contains object oriented mathematical models of aircraft flight dynamics (aerodynamic and other characteristics can be variated among a set of different planes), virtual reality models, and special algorithms for pilot's support system during maneuvering idea realization.

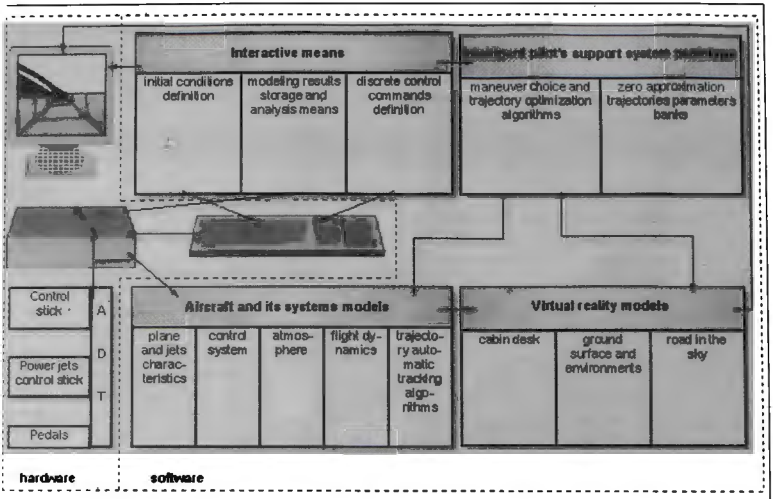
Flight dynamics equations are integrated by forth power method with step of 0.01s, display frames changing frequency is about 15Hz. In this case Pentium/133-processor makes available real time modeling.

Virtual reality graphics supplies outside and inside cabin environments visualization. 16MB RAM permits to have some hundred polygons 5kmx5km of 10 types with different texture. It's used also the idea of infinite world.

Fig.2-4 show an examples of virtual reality images.

Pilot's motor actions assisting system prototype

In order to help to operator in accepting the right decision concerning concrete maneuver formation and its subsequent performing in accordance with the well known concept[1] and pilot's requirements[2] minisimulator was added by pilot's motor actions support system[3-6] demonstration prototype.


Interactive means
initial conditions definition
modeling results storage and analysis means
discrete control commands definition
Intelligent pilot's support system prototype
maneuver choice and trajectory optimization algorithms
zero approximation trajectories parameters banks
Control stick
Power jets control stick
Pedals
A
D
T
Aircraft and its systems models
plane and jets characteristics
control system
atmosphere
flight dynamics
trajectory automatic tracking algorithms
Virtual reality models
cabin desk
ground surface and environments
road in the sky
hardware
software

ADT - analog-digital transformers

Fig.1 Block-scheme of flight minisimulator

Fig.2 An example of ground based objects visualization

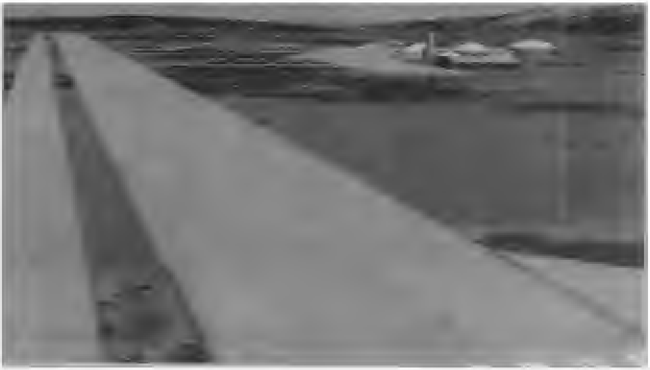


Fig.3 An example of airport environment visualization

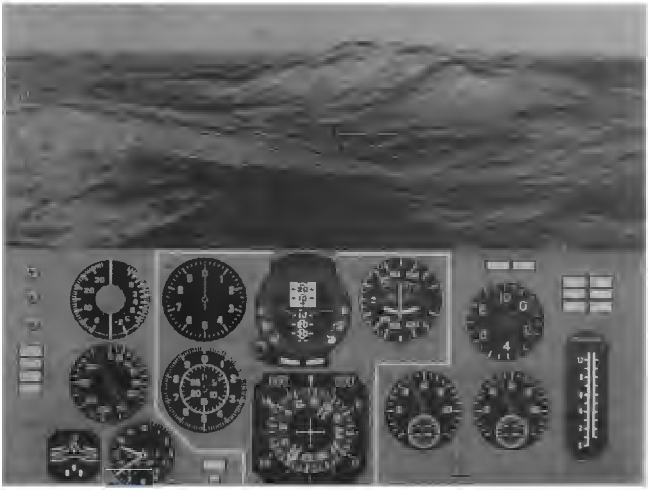


Fig.4 An example of panel desk visualization

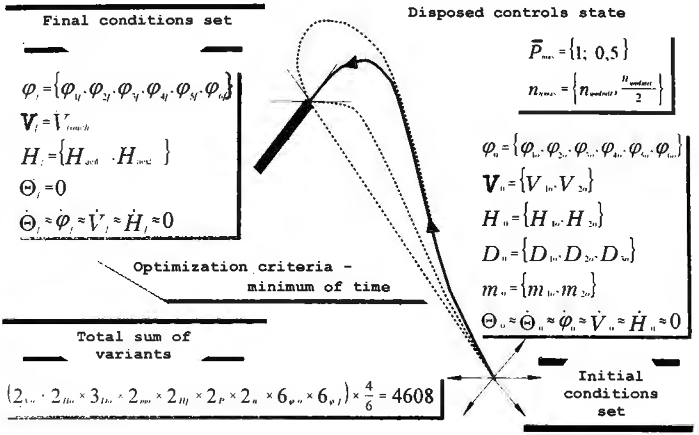

Final conditions set
$\varphi_f = \{\varphi_{1f}, \varphi_{2f}, \varphi_{3f}, \varphi_{4f}, \varphi_{5f}, \varphi_{6f}\}$
$\mathbf{V}_f = V_{touch}$
$H_f = \{H_{set}, H_{set}\}$
$\Theta_f = 0$
$\dot{\Theta}_f \approx \dot{\varphi}_f \approx \dot{V}_f \approx \dot{H}_f \approx 0$
Optimization criteria - minimum of time
Total sum of variants
$(2_{V_0} \cdot 2_{H_0} \times 3_{D_0} \times 2_{m_0} \times 2_{H_f} \times 2_P \times 2_n \times 6_{\varphi_0} \times 6_{\varphi_f}) \times \frac{4}{6} = 4608$
Disposed controls state
$\bar{P}_{max} = \{1;\ 0,5\}$
$n_{y\,max} = \{n_{y\,admit}, \frac{n_{y\,admit}}{2}\}$
$\varphi_0 = \{\varphi_{10}, \varphi_{20}, \varphi_{30}, \varphi_{40}, \varphi_{50}, \varphi_{60}\}$
$\mathbf{V}_0 = \{V_{10}, V_{20}\}$
$H_0 = \{H_{10}, H_{20}\}$
$D_0 = \{D_{10}, D_{20}, D_{30}\}$
$m_0 = \{m_{10}, m_{20}\}$
$\Theta_0 \approx \dot{\Theta}_0 \approx \dot{\varphi}_0 \approx \dot{V}_0 \approx \dot{H}_0 \approx 0$
Initial conditions set

Fig.5 An example of descent and landing task preliminary formalization

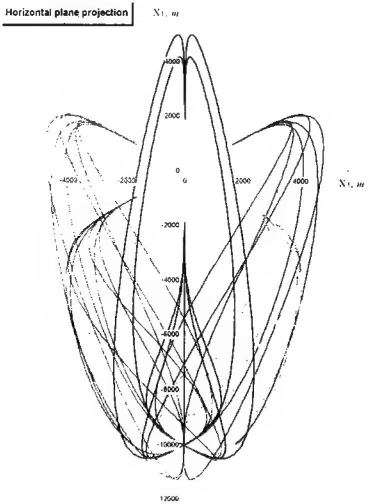
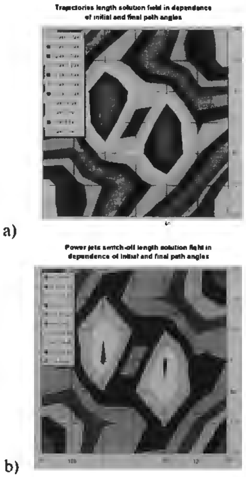


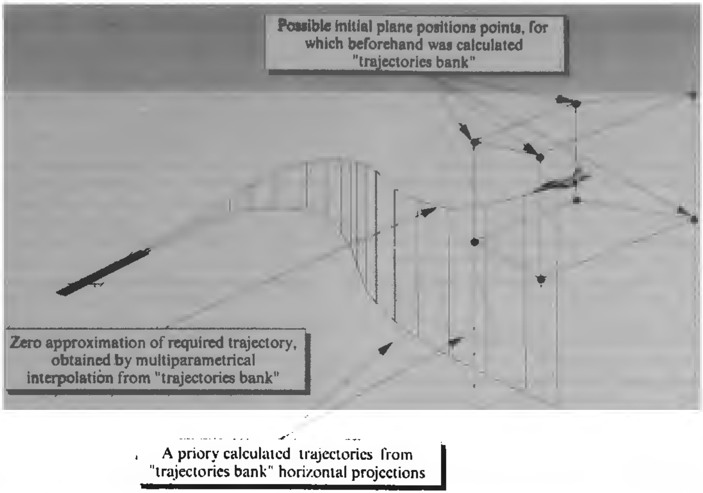
Fig.6 Fragment of optimal trajectories bank



a)

b)

Fig.7 Trajectories bank parameters fields in dependence of initial (a) and final (b) relative path angles example



Possible initial plane positions points, for which beforehand was calculated "trajectories bank"

Zero approximation of required trajectory, obtained by multiparametrical interpolation from "trajectories bank"

A priory calculated trajectories from "trajectories bank" horizontal projections

Fig.8 Zero approximation of searching trajectory from "trajectories bank" illustration

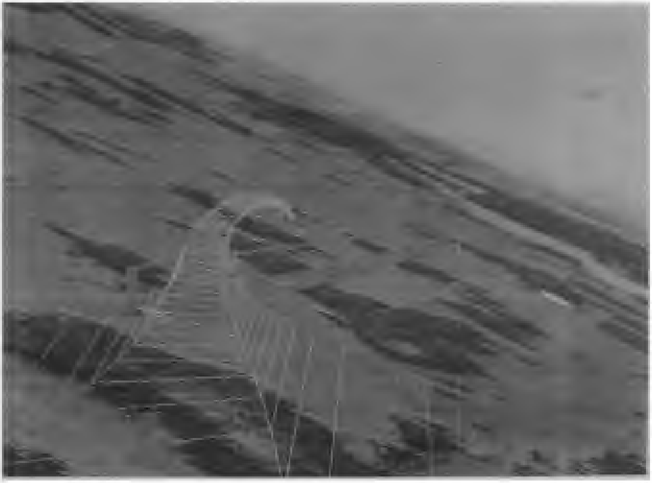


Fig.9 "Road-in-the-sky" visualization during its manual tracking example

This prototype based on beforehand calculated[*] banks of zero approximation trajectories for typical maneuvers. This trajectories defined in the nodes of a priory formalized tasks (as it shown for example on Fig.5-7 for the task of landing).

During modeling in arbitrary moment of time[&] it is given the command for optimal trajectory reconstruction. Final conditions are chosen by help of knowledge base of production type reasoning from current tactical situation. As initial conditions it is taken predicted on 2s ahead aircraft state. For this values it is made multi-parametric interpolation from trajectories bank (see graphical illustration on Fig.8), and with this good zero approximation final optimization (which required just no over 2s) is performed.

As the methodological base for variational boundary tasks solutions - both for the preliminary trajectories banks formation and operative final optimization during maneuvering (simulation) - the original modification of direct method[3] designed by author is used. Pentium/133-processor oriented algorithms permit to perform final optimization (with good zero approximation from trajectories bank) in real time - it needs only 1% of maneuver performing time.

Mentioned trajectories banks fragments for the typical tasks of ground based target strike and landing were calculated beforehand. The set of nodes was limited by 384 variants for both banks (full minimal volume of appropriate banks is evaluated as ~31000 and ~7000 zero approximation trajectories). Each trajectory is defined only by 3 parameters which are kept in relation type database.

Reconstructed recommended trajectory then is visualized on display for its tracking in manual or automatic control mode in a view of gutter (cross section of which is like lying on the left side opening square bracket[#]). Road is equipped by appropriate wayside signs like "Power-up", "Power-down", "Fire", "Drag flaps", "Gears", etc.

Fig.9 shows an example of such visualization.

In present time designers of minisimulator with prototype of pilot's support system during maneuvering are working under its full-scale test and further development concerning possibility to assist to operator in more wide set of performing tasks.

References

1. *Stein K.J.* DAPRA stressing development of pilot's associate system. Wright Laboratories Broadens Advanced Technologies Initiatives // Aviation Week, 1985, vol.122, №16. - P.69-81.
2. *Yakimenko O.A.* Pilot requirements to universal airborne intelligent pilot decisions support system // Proceedings of NAECON-95. - Dayton, 1995.
3. *Yakimenko O.A.* The airborne intelligent pilot motor actions decisions support system and its methodological foundation // Proceedings of NAECON-95. - Dayton, 1995.
4. *Yakimenko O.A.* "Road in the sky" as the means of pilot's motor action during maneuvering intelligent support // Proceedings of NAECON-96. - Dayton, 1996.
5. *Theunissen E.* Factors influencing the design of perspective flight path displays for guidance and navigation // Displays, 1994, vol.15, №4. - P.241-254.
6. *Theunissen E., Mulder M.* Open and closed loop control with a perspective tunnel-in-the-sky display // A collection of technical papers of AIAA-94 Flight Simulation Technologies Conference, Scottsdale, 1994. - P.32-42.

[*] By use of direct method modification (see below).

[&] The only limitation is the accessory of aircraft relative phase coordinates to bank trajectories definition region.

[#] Because this type was chosen by pilots as most suitable one[2].

AIAA-96-3551-CP

IMPORTANCE OF SPATIAL SOUND CUES IN S IMULATORS

Arnab Lahiri
Staff Systems Engineer
NATCO, Eagan, MN

Abstract

Flight simulators attempt to replicate an environment conducive to aircrew training . The designated level of the simulation is a directly dependent on the resultant fidelity towards realism. This fidelity has been the key factor that has enabled progression in transfer of training both in procedures and methods. Amongst the many different cues available sound plays a very important role in a simulator training scenario. With upcoming regulations governing the fidelity of flight simulators, the status of simulated sound has been further elevated. Sufficient emphasis has, however, not been placed on the spatial aspects and effects of sound. Sound cues, especially in the cockpit, does not only alert to a single or a series of events but also attempt to provide various information regarding the source(s). Use of modern sound recording techniques can be easily utilized along with multiple sound gathering elements to obtain the kind and quality of sound data needed. These sounds can then be duplicated in a multi-channel environment for closer spatial simulation. Testing would involve the use of similar setups, though in a much simpler form which would also ensure repeatability. Simpler setups would reduce cost and time involved as well. It is necessary to justify such fidelity which would ultimately ensure not only enhanced simulation to further training requirements but reasonable methods of maintaining the integrity of the standards of the sound as well.

Background

A few decades ago, the flight simulators of today were still in their infancy. Simulators were based on cues an as long as the trends were present in the cues, the simulation was considered to be adequate. Training procedures and credits granted were geared towards the level of simulation that was available. The level of simulation as per regulatory demands was dictated by the degree of fidelity that the simulation model would deliver. Simulation progressed as more accuracy in the aerodynamic modeling were developed and motion and visual systems response were enhanced for better realism. A quick review of the history of flight simulators will reveal that among the various elements in any simulation, very little importance importance was placed on aural cues. Sound systems were easier to work with and hence the final simulated output was left to the discretion of the individual engineer. Even after the institution of governmental regulations to standardize the fidelity of flight simulators, sound was still considered to be a purely subjective element. Very little attempt was made to incorporate accuracy into the modeling. However in recent years with the advent of higher levels in the degrees of simulation, the industry has begun an attempt to better quantify the sonic fidelity in flight simulators. The question still remains whether this level of fidelity is absolutely necessary and whether the cost will ultimately contribute significantly towards better training.

Sound Cues

Sound cues are of various nature. Inside the cockpit the sounds can be from various sources. There are distinct sounds both discrete and periodic from the systems, the power units or the environment. Being a very complex system, the airplane has a vast array of events which can occur simultaneously. And each event produces one or more sensory information which may need to be processed by the crew. Monitoring all of this becomes a major task. Thus if more sensory sources are available the process of identification of the event hopefully becomes easier. Ofcourse there are instances where more may cloud judgement. To the flight crew inside the cockpit reliance is placed on every bit of sensory information source ranging from visual, tactile and auditory. More than often a combination of these reinforce and confirm the facts of an event. It is important to note that simulated aural fidelity like most other sensory stimuli in the simulator is often very dependent on the spatial dimentionality of the source. From a tactile standpoint, motion, control feel and vibrational cues are all dimensional. The information often gleaned from these cues are the nature of the event and its location. By selectively narrowing the information it becomes possible to establish the cause, gauge its effects and take the appropriate action. It is the same for the visual aspect. Therefore it stands to reason that aural cues do also provide very similar results.

The Spatial Effect

A proposition often forwarded is that sound plays a lesser role in current training requirements than say the visual or motion or control feel. For what we are allowed to achieve in training under the current regulations, the fidelity of the audio in the simulator is adequate. One has to only look back to find that similar arguments were and still are used against other simulated cues as well. Regulations always do not cover the total entity that is the simulator. Enhancing the fidelity beyond the regulations especially in the training and learning environment is always a plus. Training is not just learning the rules but developing the senses and judgment to react based on all the information that is made available.

Most modern flight simulators do have some sort of spatial effects in their audio systems. However the accuracy of these effects are never established. Speaker placements are often arbitrary and multi-channel audio systems are used to produce some of the more noted effects. Normal and abnormal sounds are duplicated with very defined assumptions. If the sound source is identified as emanating from a certain direction, the simulation inside the cockpit is executed as it were from the same direction. Little accounting is made for the fact that the source may be affecting other elements of the system in the real world to produce a dimensionality in both intensity and quality which may be quite removed from the original assumption. Under current regulations the testing procedures the methods used for evaluatory purposes do not allow for the confirmation of the spatial aspect. The spatial effect is still purely subjective which at once involves errors in dimensionality.

Testing

Testing procedures are essentially quite simple. The simulated sound is captured at a pre-determined point, usually at the pilots right ear position inside the cockpit. Calibrated microphones and recorders are used to capture the sound produced. The captured data is then mathematically analyzed to produce graphical results which are compared with the original analyzed data from an aircraft. The resultant comparison provides no clues as to the directionality of the sound sources. Essentially the comparison is single dimensioned. Due to the higher level of ambient noise inside the simulator because of functioning equipment, the sound needs to be both distinct and pronounced. A better procedure, ideally would be to use three simultaneous audio capturing elements placed around standardized baffles to simulate the seated pilot. This kind of analysis would give a far better picture of the fidelity of the sound including its spatial aspect. An immediate concern is the justification of the cost of equipment and the time involved in such analyses. Some reductions in the analyses are always possible without sacrificing too much fidelity. There are sounds in an airplane which do not require the spatial simulation in order to achieve the desired results. It is possible to identify the more important sound which have directionality.
A stereophonic sound gathering set up would be the most practical compromise reducing cost and time. The analysis for the spatial sounds would entail comparison of the intensities and content of the frequencies in each direction. This would indeed enhance the fidelity of the simulation considerably.

Conclusion

The final realism in a simulator is always a combination of effects. For the purpose of modeling, various aspects of the simulation are mathematically handled as separate entities. Each entity has in the past been individually enhanced to be more representative. Equal degrees of progress has not been always instituted to each of these individual aspects and some have lagged in realism and fidelity- sound is one. Bringing further fidelity to the level in the sound quality contributes to the entire package of the flight simulation and this can only further training needs. Better perception will eventually lead to better recognition.

References

1. Rolfe, J.M. & Staples, K.J.; Flight Simulation, Cambridge University Press, 1989

2. FAA Advisory Circular AC 120-40B; dated June 9, 1993.

3. McCormick, E.J.; Human Factors Engineering, McGraw-Hill Book Co., 1970.

AIAA-96-3552-CP

TOWARDS A CONCEPTUAL FRAMEWORK FOR COMPARING PERSPECTIVE FLIGHTPATH DISPLAYS

Erik Theunissen, Delft University of Technology, Faculty of Electrical Engineering
P.O. Box 5031, 2600 GA Delft, The Netherlands (e.theunissen@et.tudelft.nl)

ABSTRACT

The conventional instrument for precision manual control is the flight director command display. Another option, which is the basis of the perspective flightpath display, is to present the guidance requirements in such a way that they can be used for control. The numerous options for the representational aspects and the selection of values for the design parameters yield an enormous variety in potential display formats. No detailed guidelines exist to translate navigation task requirements into a specification of representation and design parameters. For a structured approach to the design of a perspective flightpath display, a framework integrating perceptual, cognitive, and control-theoretical aspects is needed.

When evaluating different display formats with pilot-in-the-loop studies, measures such as lateral and vertical tracking performance only provide an indication of how well the pilot is able to track a forcing function and might create a bias toward command displays. In reality, the pilot has to remain within certain constraints rather than exactly stay in the middle between them. To allow a fair comparison of different display formats for aircraft guidance, measures should be included which allow an evaluation of the full range of potential control strategies to satisfy task requirements.

In this paper, an approach to a design framework is presented and a task and parameter are proposed to measure the pilot's ability to utilize the information about the constraints rather than the information about the forcing function.

1 INTRODUCTION

To safely accomplish the navigation task, pilots need to be aware of the aircraft position and orientation relative to the desired trajectory. The current navigation display provides the pilot with a planar picture of the aircraft's position and heading relative to the flightpath. It significantly lacks in the ability to provide navigational awareness in all three spatial dimensions. Other displays provide information about the vertical dimensions. Achieving an adequate level of navigational awareness in all three spatial dimensions necessitates a lot of scanning and mental integration, increasing task demanding load as a result of non-optimal displays. The anticipated increase in traffic density and the desire for flexible, curved approach paths will further increase task demanding load, especially in situations where it is already very high.

- To avoid that future developments impair safety, better navigation and guidance displays must be developed which require less effort from the pilot to stay on top and ahead of the situation.
- To reduce the sudden build-up of task demanding load, displays should provide information which enables pilots to operate in an open-loop mode allowing anticipation of future events.

Navigation, guidance, and control are not three independent tasks. To maintain a high degree of cognitive coupling between the tasks, displays should reflect the relation between them. The idea behind current guidance displays is to isolate the control task and treat the pilot as a servo mechanism. This is an enormous waste of the capabilities of the human operator. Rather than spending the major part of his resources to behave like a good servo, the role of the human operator should be to compensate for the limited flexibility of the automated systems. The displays have to provide the information the pilot needs to monitor and anticipate the situation as it develops, and intervene with maximum efficiency when necessary.

- The elements of the display which provide guidance should not force the pilot to function as a servo and apply a continuous compensatory control strategy. Rather than commanding the pilot what to do, or at best showing only the error with respect to the desired trajectory, guidance and navigation displays should provide information about the margins within which the pilot is allowed to operate. Only in this way can human flexibility be exploited. This is a fundamental difference with current command displays.
- Well designed displays providing information about the constraints within which the pilot is permitted to operate, allow a trade-off to be made between workload and performance. For the control task this implies that the display should provide the information the pilot

needs to apply anticipatory and error-neglecting control strategies.

Perspective flightpath displays have the potential to satisfy these requirements. Although having been discussed for over forty years, the flight director command display is still the only instrument used for precision manual flight. Until about a decade ago, technology was the limiting factor for the implementation of perspective flightpath displays. Now, this is no longer the case and the reasons for employing less sophisticated displays (which are based on the technical possibilities of forty years ago) must be revisited.

- One should always consider that every existing implementation is a trade-off which resulted from the technical limitations of the time it was designed. Therefore, it is important to understand why a certain design was selected and others rejected.

Whereas the development of a flight director command display is mainly a control engineering problem for which structured approaches to the design exist, a perspective flightpath display requires consideration of control theoretical, perceptual and cognitive aspects. In contrast to guidelines for flight director design, no detailed design guidelines exist for perspective flightpath displays.

Nine years ago, Fadden et al.[3] stated that *"While the promise of spatial displays is great, the cost of their development will be correspondingly large. The knowledge and skills which must be coordinated to ensure successful results is unprecedented. From the viewpoint of the designer, basic knowledge of how human beings perceive and process complex displays appears fragmented an largely unquantified"*.

Since that time, research into the perception and control of self-motion has increased the understanding of how humans perceive and process information from spatial displays. Research into various aspects of perspective flightpath displays dates back to the early fifties. The numerous options for the representational aspects and the selection of values for the design parameters yield an enormous variety in perspective flightpath display formats. The different concepts which have originated over the past forty years can only be compared with each other in terms of design parameters.

A clear need for a structured approach to the design of perspective flightpath displays exists. When making modifications or proposing new designs, it is of crucial importance to understand the motivations which resulted in the current and past ones and the reasons which caused other approaches to be abandoned. A design framework is needed to make and justify design decisions in the context of task requirements based on existing human factors knowledge and findings from previous research. Such a framework also allows one to determine plausible causes for problems and can serve as a guide towards solutions. Its development is being pursued in the context of the Delft program for hybridized instrumentation and navigation systems phase 2 (DELPHINS II). As part of this program, guidelines to the representation of the flightpath and the specification of design parameters have been developed. These guidelines relate design decisions to the effect they will have on the task-specific visual cues that are conveyed by the display. As such, they also allow an evaluation of previous designs, thus supporting the integration of findings from other research into perspective flightpath displays. Since the limited length of a paper does not provide the opportunity to present a detailed overview of all aspects involved in the design, this paper focused on providing an overview of the approach which is being used to develop design and evaluation guidelines.

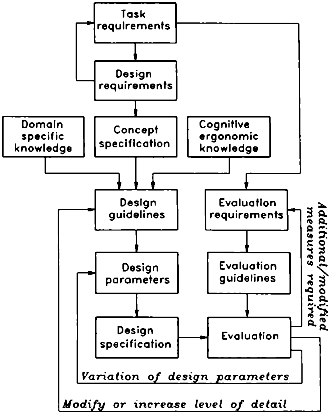


Figure 1 Specifying and refining design guidelines

2 APPROACH

Figure 1 presents an overview of the process of creating a design framework as being used in DELPHINS II. Initial design guidelines based on an integration of existing knowledge are far from complete. Additional information must be obtained through an iterative process involving domain experts and end-users, in which each cycle serves to increase the level of detail of the design framework. To refine design guidelines, hypothesis about the influence of design parameters on task strategies and/or performance are made. Through variation of the design parameters and successive evaluation, the hypothesis are tested.

2.1 Design guidelines

To make and justify design decisions, an understanding of the relation between the representational aspects, the design parameters and the task related variables is needed.

The representation determines the perceptual and cognitive effort which is needed to translate the perceived image into relevant information. Section three focuses on the representational aspects.

To determine the influence of design parameters on task variables, the relation between visual cues and potential control strategies which are possible to satisfy task requirements is needed. This is discussed in Theunissen and Mulder[18]. The identified visual cues must be further divided into specific properties of the optic flow field. Results from research into the perception and control of self motion have provided an organizational framework which describes the relation between parameters of an optical flow pattern and control actions[11]. By relating the control actions to specific task requirements and describing the magnitude of the visual cues conveyed by a perspective flightpath display as a function of the design parameters, a better understanding of the influence of changes in the design parameters on task performance can be obtained. Section four focuses on the description of task related visual cues as properties of the optic flow field.

2.2 Evaluation guidelines

Flight director displays are compared by using measures which indicate how well the pilot is able to track a certain forcing function. These measures provide no indication of the pilot's ability to apply other control strategies which, for example, provide an opportunity to make a trade-off between workload and performance.

To allow a fair comparison of different display concepts for aircraft guidance, measures should be used which allow an evaluation of the full range of potential control strategies to satisfy task requirements. Thus, besides comparing display concepts in terms of tracking performance and control activity, there is a need for measures which indicate the ability to apply other control strategies than compensatory control. In section five, a task and a measure are proposed to quantify the pilot's ability to extract information about temporal margins as a function of display.

3 REPRESENTATION

Task demanding load is influenced by the representation. Wickens[21] states that "*At any level of perceptual processing it should be apparent that the accuracy and speed of recognition will be greatest if the displayed stimuli are presented in a physical format that is maximally compatible with the visual representation of the unit in memory*". An important feature of humans is that certain compatibility relations, such as the spatial correspondence between stimulus and response seem to be intrinsically related to the 'hard wiring' of the nervous system. If such features can be exploited, the cognitive load can be minimized. The ideal display format is one in which the visual stimuli convey the required information so that the processing for perceptual encoding is minimized. Each level of stimulus dimension may be referred to as a feature[21]. A perspective flightpath display format can be described as a set of components, in most cases line segments, each having specific features such as length, orientation, thickness, and color. Together these components form an object, the tunnel, which itself also has features, one of which is symmetry. To bypass the feature analyzers on the component level, the flightpath should be presented in such a way that it allows holistic perception. In this way, the pilot does not have to decompose the object into features which are relevant for the task. Rather, the emergent features of the object are directly perceived, and can be used to initiate natural multi-dimensional control responses.

The integration of the third dimension causes ambiguity. Ambiguity can only be resolved with additional information. Sometimes this additional information is not explicitly available and assumptions about the structure of the environment are used. In case of an erroneous assumption, misperception results. To prevent errors resulting from misperception, it is important to understand its causes. When dealing with potentially misleading visual cues in the real world, adequate training can be used to compensate. In contrast to the real world, the designer of a perspective flightpath display has absolute control over the structure of the virtual environment which conveys the visual cues, and as a result has more possibilities to prevent misperception through adequate design.

- The representation should contain elements which provide the cues required to minimize ambiguity and avoid misperception resulting from the integration of the third dimension.

3.1 Shape

The representation should stimulate holistic perception, which requires the presence of visual cues defining the shape of the object. The fact that even in case the contours are not physically complete our perceptual mechanism completes the contours through top-down processing, presents the designer with some freedom which can be used to make a trade-off between the available computing power, amount of detail and potential display clutter.

Since the detection of symmetry takes place in the early processing cycles of visual information, this feature can be exploited to reduce the required effort for interpretation and evaluation. The shape of the flightpath and the frame-of-reference should be selected so that they provide the observer with the cues to determine position and orientation errors and exercise control to keep the errors within predefined constraints. Any other frame-of-reference than an egocentric one cannot exploit the advantage of a symmetrical reference condition, and will require additional mental processing.

- To minimize cognitive processing for the guidance task, the object which represents the flightpath should be symmetrical around the horizontal and vertical axis, and an egocentric frame-of-reference should be used.

Rectangular shaped cross-sections provide information about horizontal and vertical position constraints. The height and width of the cross-sections determine the position error gains. Sometimes, it is desirable to also have a source of a very high position error gain which can be used for temporary fine-tuning. Reducing the tunnel size to obtain this high gain would force the pilot to continuously apply a high control gain. By presenting references indicating the center of the tunnel sections, additional cues providing direct information about the deviation from the center are available. In fact, the altitude poles already provide such information for lateral control. During experiments[15], pilots mentioned that in the final approach they used the alignment of these poles for accurately positioning the aircraft on the centerline. An alternative to rectangular shaped cross-sections is to use diamond shaped ones. Rectangular elements, however, have the advantage that in the absence of errors the lines of the cross-sections run horizontal and vertical, and that judgement of horizontalness and verticalness is something people are very good at. Some representations are not symmetrical around the horizontal axis and thus do not provide altitude cues by means of an emergent feature. As a result, additional symbology is integrated to provide the pilot with these cues. Table 1 presents an overview.

Ref.	altitude cues
Wattler and Mulley[19]	lead plane vertical position
Knox and Leavitt[10]	reference shadow on path
Filarsky and Hoover[4]	lead plane vertical position
Reising et al.[14]	lead plane vertical position

Table 1 Additional altitude cues

When solid lines are used to indicate the connections between the cross-section frames, as is the case with many implementations of perspective flightpath displays, the only velocity cues are those available from the motion of the cross-section frames. When using a representation of the tunnel with dashed interconnections between the cross-sections, velocity cues can be significantly increased.

The orientation of the tunnel can be used to provide the pilot with additional information. Grunwald[8] describes a method to generate a tunnel in which the elements are banked in curves and transitions to curves, providing roll commands for carrying out a coordinated turn at a given velocity. Whereas the position constraints indicated by the tunnel wall provide outer-loop cues, the reference bank angle is an inner-loop cues. Matching the actual roll angle with the commanded roll angle does not necessarily guarantee that the outer-loop requirements are satisfied.

3.2 Ambiguity

Altitude poles and a ground track relate the flightpath to a ground plane and can reduce ambiguity. The altitude poles also provide a possibility to temporarily use a very high lateral error gain. Besides stereopsis, certain velocity cues are considered to very strongly convey a feeling of three-dimensionality and resolve ambiguities. As a result of the apparent motion of the cross-section frames towards the observer, and the resulting optic flow field, the feeling of three-dimensionality increases, and ambiguities are further reduced.

- To resolve ambiguities, both the representation and the design parameters must be selected in such a way that adequate velocity cues are conveyed. Since the magnitude of these cues is proportional to ground speed, this is only possible over a certain range of velocities.

3.3 Misperception

As indicated previously, velocity cuing can be increased by using dashed lines rather than solid lines to indicate the interconnections between the boxes. A fundamental requirement is that the spacing between the dashed lines in the 3-D world is correctly transformed to the 2-D presentation. Just connecting the boxes with dashed lines with equal 2-D spacing generates a constant edge-flow pattern, but conveys conflicting global optical flow cues. Grunwald[8] describes a display format which uses tunnel elements with a length of 200 ft spaced 200 ft apart which yields both accurate edge-rate and global optical flow cues. Misperception of velocity can also result in case one of the factors determining edge-rate and global optical flow-rate changes while the observer is unaware of it. A tapered segment of the perspective flightpath which might be needed to gradually increase the position constraints also increases global optical flow-rate and can potentially yield a misperception of velocity. In case the frame-spacing decreases, the increase in edge-rate might cause the observer to misperceive this as an increase in velocity. As indicated by Owen[11] experimental evidence suggests that sensitivity to edge-rate and global optical flow-rate cues varies among people. Furthermore, all velocity cues conveyed by the representation of the flightpath are relative and inertially referenced.

- Since the velocity cues resulting from the dynamic presentation of the flightpath cannot be considered reliable indicators for either absolute or relative velocity and are inertially referenced, additional data must be presented

4 TASK RELATED VISUAL CUES

Theunissen and Mulder[18] discuss the different types of control strategies which are possible as a result of the visual cues conveyed by a perspective flightpath display. Each control strategy requires the presence of certain cues, and the resulting control behavior and performance are determined by the magnitude and the resolution of these cues. In order to compare different designs, the specific characteristics of the cues for each control strategy must be described as a function of the design parameters. Based on the different types of control strategies, it is possible to distinguish between cues needed for compensatory control, cues needed for anticipatory control, and cues needed for error-neglecting control. All three categories and the relation between the magnitude of the cues and the design parameters will be briefly discussed, and it is indicated how the cues can be expressed as properties of the optic flow field.

4.1 Describing cues for compensatory control

To describe the visual cues for a compensatory tracking task in the presence of trajectory preview, but in the absence of changes, the effect of the trajectory preview is that a single snapshot of the presentation contains information about both position and orientation errors. Since the latter ones are proportional to position error rate, the preview integrates position and position rate information into a single snapshot.

To better understand the contribution of the visual cues to task performance, a description of the functional variables for the specific task(s) is needed. Theunissen[16] presents the relation between the distortion of symmetry, position and orientation errors, tunnel size, and geometric field-of-view. The parameters in the equations expressing the distortion of symmetry are directly related to the 3-D world. For every element of the tunnel which is presented on the 2-D display, distance to the viewpoint must be known to calculate the distortion of the symmetry caused by that specific element. Although the equations are useful to provide more insight in the contribution to position and orientation cues of elements at a certain spatial location, the third dimension in these equations is not useful for relating the perceived visual cues to the control actions. Since the emergent feature (distortion of symmetry) is a 2-D phenomenon, i.e. the magnitude of the distortion does not vary along the viewing axis, an expression relating the distortion to position and orientation errors as a function of tunnel size and field-of-view without including the third dimension is desirable. In order to relate control actions to the available visual cues in the optic flow field, such an expression should directly relate the position and orientation of the elements in the 2-D representation to the position and orientation errors. An additional advantage of expressing the effects of spatial position and orientation errors as 2-D cues is that perceptual thresholds can be related to minimum perceivable differences in spatial position and orientation errors. This allows the designer to specify minimum display size and resolution.

The effect of orientation errors can be approximated by a translation of the displayed image. For small position errors, the distortion of symmetry can be approximated by a rotation of the tunnel lines. Figure 2 illustrates how a change in lateral position causes a rotation of the tunnel lines. Owen[11] refers to these angles as *splay angles*. With a perspective flightpath display presenting a tunnel of width w and a height h, the splay angle S of the tunnel lines in the absence of position errors is equal to:

$$S = \arctan(\frac{w}{h}) \qquad (1)$$

Horizontal and vertical translations of the viewpoint result in changes in the splay angle. When the position errors are small compared to the tunnel size, the changes in splay angle are proportional to the position errors and can be approximated through:

$$S_{XTE}=\frac{XTE}{w} \qquad [Radians] \tag{2}$$

for lateral position errors, and

$$S_{VTE}=\frac{VTE}{h} \qquad [Radians] \tag{3}$$

for vertical position errors. Owen[11] describes several studies which all indicate that splay rate is the functional variable for altitude control. A symmetrical object such as the tunnel provides splay rate cues both for vertical and horizontal position control. It is therefore assumed that with a perspective flightpath display, splay rate is the functional variable for position control. Based on Owen's organizational framework, in the middle range of sensitivity an equal-ratio increment in splay gain should provide an equal-interval improvement in performance. This trend is indeed observed in an experiment which investigated the influence of tunnel size on tracking performance[15], strengthening the hypothesis that with a perspective flightpath display splay rate is a functional variable for position control.

- By describing the control oriented visual cues as a distortion of the symmetry, they can be divided into changes in optical splay angle, scene translations, and scene rotations. With this approach, it is possible to compare different perspective flightpath displays with each other in terms of task-related visual cues and the magnitude of these cues.

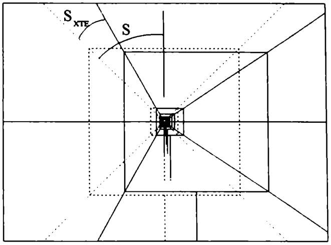


Figure 2 Influence of a position error.

The elements representing the cross-sections of the flightpath can be scaled in their geometric dimensions, and thus determine splay gain. The geometric field-of-view determines the amount of scene translation caused by a change in orientation. Without pilot models which describe control behavior as a function of the visual cues conveyed by a perspective flightpath display, the determination of suitable values for splay gain in order to satisfy tracking performance requirements is an iterative process requiring pilot-in-the-loop experiments. Based on the assumption that in the middle range of sensitivity, an equal-ratio increment in splay gain yields an equal-interval improvement in performance, it is possible to rapidly determine whether desired performance can be achieved by selection of the splay gain, or that display augmentation, e.g. by means of predictive symbology is needed. To provide an indication of possible values, Table 2 lists the horizontal splay-gain G_H (degrees/m), the vertical splay-gain G_V (degrees/m) and the translation gain G_T (percentage of screen displacement per degree of rotation) which have been calculated for several studies into perspective flightpath displays.

Ref.	G_H	G_V	G_T
Grunwald et al.[6]	1.25	1.25	1.1
Grunwald[7]	0.42 0.63	0.42 0.63	1.1
Way et al.[20]	0.63	1.25	4.3
Wickens et al.[22]	2.28	3.35	2.8
Barfield and Rosenberg[1]	0.94	0.94	1.3 1.7 2.2
Dorighi et al.[2]	0.63	0.63	1.1
Theunissen[15]	0.64 1.27 2.55	0.64 1.27 2.55	1.9

Table 2 Splay and translation gain

One could argue that a comparison of the design parameters provides the same information, which is true, but an overview as presented in Table 2 directly shows the difference in magnitude of task related cues rather than just a difference in design parameters.

4.2 Describing cues for anticipatory control

Anticipatory control is of importance when transitioning between straight and curved segments. To apply this control strategy, the pilot must be able to estimate the moment of initiation and the required magnitude of the anticipatory action based on the preview. Theunissen and Mulder[18] discuss the specific cues in more detail and conclude that it is difficult to estimate the required magnitude of the anticipatory control action. With respect to temporal range cues, research performed by Kaiser and Mowafy[9] indicates that the accuracy with which time towards objects can be estimated depends on the time between the moment the objects leaves the viewspace and the moment the object passes the viewplane of the observer. For the cross-sections of the perspective flightpath displays, this time is a function of relative velocity, geometric field-of-view, and tunnel size. Theunissen and Mulder[18] refer to this time as the minimum time to passage (TTP_{min}). When comparing different formats, TTP_{min} can provide an indication of the possibility of extracting useful temporal range information.

4.3 Describing cues for error-neglecting control

A presentation of data which allows the pilot to determine the current and future trajectory relative to the current and future position constraints, provides the opportunity to make a trade-off between control related workload and tracking performance. Therefore, it is important to understand which cues contribute to the pilot's ability to estimate the available margins and willingly ignore certain position and orientation errors. Theunissen and Mulder[17] introduced the so-called time-to-wall crossing (TWC) parameter which represents the temporal distance towards the tunnel wall. The TWC concept is based on the assumption that there is a relation between the remaining time the aircraft is within the boundaries indicated by the tunnel walls, and the moment an error-corrective control action is initiated. Results from a pilot-in-the-loop study into error-neglecting control behavior[17] strengthen the assumption underlying the TWC concept.

Until now, design parameters have been related to tracking performance which can be achieved[7,15] when applying a continuous compensatory control strategy. As pointed out in the introduction, displays should provide information about the margins within which the pilot is allowed to operate, allowing an error-neglecting control strategy.

- A method which allows the qualification of the ability to extract information about margins in terms of temporal range as a function of the design parameters would provide the opportunity to consider this control strategy during the specification.

5 EVALUATION GUIDELINES

In general, evaluations are performed at several phases during the design and can range from a numerical closed-loop analysis to pilot-in-the-loop experiments. Numerical analysis can be used to gain more insight into system performance and stability as a function of the design parameters.

5.1 Numerical closed-loop analysis and pilot models

For numerical closed-loop analysis, all elements in the loop must be described, thus a pilot model is required. For a flight director tracking task, validated models are available but unfortunately this is not (yet) the case for perspective flightpath displays. The fact that perspective flightpath display allows a much larger range of control strategies to be applied, makes the task of modeling the control behavior of the pilot much harder.

- The flexibility in response caused by the presentation of status rather than command information increases the variability in pilot performance. This may necessitate new approaches to model the contribution of the human element. The required level of detail of a pilot model has yet to be determined.

A specific challenge lies in the development of a pilot model which takes the anticipation of changes in the future forcing function as presented by a perspective flightpath display into account, and thus allows a prediction of both the anticipatory open-loop and the compensatory closed-loop control actions based on the cues presented by the display. With such a model, the need for additional symbology to aid the pilot in determining the timing and magnitude can be established based on system dynamics and display design parameters. Simplified pilot models which describe the pilot's compensatory control behavior can be very useful for predicting performance and system stability as a function of changes in the design parameters.

A simplified model can be based on the assumption of continuous compensatory tracking. Grunwald and Merhav[5] showed that the velocity field resulting from the relative motion between the viewpoint and the 3-D environment contains information about the future position error. For closed-loop stability analysis, they assumed that the pilot's control actions are proportional to the perceived future position error. This approach requires an assumption to be made about the part of the future which is used by the observer. They used a 'two-distance model' to include the effect of a span of viewing distances. Whereas the basic assumption underlying Grunwald and Merhav's approach[5] is that the pilot extracts his future position error from the velocity field and uses this error to determine his control action, the discussion in section 4.1 assumed that the basic cues the pilot uses are the distortion of the natural symmetry of the tunnel and the rotation of the image. It is likely that the representational aspects influence the use of the specific cues. When a perspective flightpath display uses a continuous connection between the cross-section frames, the only velocity field cues are those conveyed through the motion of cross-section frames towards the observer. When using a representation of the tunnel with dashed lines representing the position constraints[8], velocity field cues can be significantly increased. It can be shown that mathematically the two assumptions yield similar approaches since they both relate the magnitude of the control action to a weighted combination of the current position error and its first and second derivative

5.2 Tasks and performance measures

With pilot-in-the-loop experiments, tracking performance and control activity[7,15], awareness[12], and workload[3] are the most commonly used measures. In reality, pilots will certainly not continuously maximize tracking performance nor utilize the available margins to the full extent. They apply a control strategy which is a mix of compensatory, anticipatory, and error-neglecting control. When applying this control strategy in an experiment, it becomes very difficult to investigate the relation between aspects of a certain control strategy as a function of visual cues. Therefore, tasks must be defined which force the pilot to dominantly apply the particular control strategy of interest.

When investigating tracking performance as a function of certain design parameters, the common approach is to instruct pilots to maximize tracking performance. As indicated previously, besides measures indicating tracking performance, measures are needed which allow a comparison over the full range of potential control strategies. Control strategies utilizing preview on the future forcing function can be characterized by the type of anticipatory control actions which are possible as a result of the preview. Such an action can be an open-loop input or a deliberate neglection of certain errors. For an open-loop input the pilot needs information about the required time and magnitude of the control action. For a certain change in the forcing function, an optimal value of both time and magnitude can be computed and compared to the pilot's control input.

To obtain an indication of the suitability of a display format for error-neglecting control, a task must be defined which emphasizes the use of information about position constraints. When investigating the use of cues for error-neglecting control, pilots have to be motivated to abandon continuous compensatory control and apply an error-neglecting control strategy. Although this is certainly not representative for the actual control strategy in real flight, this approach makes it possible to isolate the specific cuës which cause the pilot to intervene. In section 4.3, the TWC parameter was discussed. After an error-corrective control action is initiated, the TWC will initially decrease further until a minimum is reached (TWC_{min}). With an error-neglecting control task, both the TWC at the moment an error-corrective control action is applied (TWC_{EC}) and TWC_{min} should provide a good indication of the pilot's ability to extract information about constraints as a function of display format.

6 SUMMARY AND CONCLUSIONS:

The main difference between a flight director and a perspective flightpath display is that the former one presents control commands, whereas the latter one presents guidance requirements in such a way that they can be used for control. For flight director displays, detailed guidelines to the design exist, which is not the case for perspective flightpath displays. It was indicated that the development of design guidelines for perspective flightpath displays requires a framework in which control-theoretical, perceptual, and cognitive aspects are integrated. Such a framework requires a detailed description of the relation between the information provided by the presentation and the potential control strategies which are possible to satisfy task requirements. Studies performed in the field of experimental psychology have provided an enormous amount of knowledge about perception, cognitive processing, and control behavior of the human operator. To apply this knowledge to the design of a man-machine interface utilizing spatially integrated trajectory preview, it is necessary to translate specific design questions into a more general context. Although this requires an up-front investment, the alternative of not applying the vast domain of existing knowledge will certainly necessitate an increased number of pilot-in-the-loop experiments to answer design questions.

By describing the visual cues conveyed by the perspective flightpath display as properties of the optic flow pattern and expressing the magnitude of the cues as a function of the design parameters, results from research into perception and control of self motion can be used to provide a better understanding of the influence of changes in design parameters on task performance.

At present, the framework is by no means complete and cannot be used for a full detailed design. However, it already provides an adequate description of the relation between the multitude of design aspects and the resulting visual cues in a task related context. Therefore, it can be used to compare different concepts in terms of task related visual cues which is a significant improvement over a comparison based solely in terms of a number of design parameters. For further development of the framework, an important question is where and how the level of detail can be increased without sacrificing generelizability.

7 ACKNOWLEDGMENTS

The work described in this paper has been performed at the Telecommunication and Traffic Control Systems group (TVS) of the Faculty of Electrical Engineering at Delft University of Technology. The development of the design framework takes place in the context of the Delft Program for Hybridized Instrumentation and Navigation Systems (DELPHINS) Phase II, which runs from 1996 until 2000 and is mainly funded by the Dutch Technology Foundation (STW).

8 REFERENCES

1. **Barfield, W.B. and Rosenberg, C.** (1992). 'The Effect of Geometric Field-of-view and Tunnel Design for Perspective Flight-Path Displays', *SAE 921131*, Society of Automotive Engineers.

2. **Dorighi, N.S., Ellis, S.R., and Grunwald, A.J.** (1993). 'Perspective Format for a Primary Flight Display (ADI) and its Effect on Pilot Spatial Awareness' *Proceedings of the Human Factors and Ergonomics Society 37th Annual Meeting*, Seattle, WA.

3. **Fadden, D.M., Braune, R. And Wiedemann, J.** (1987). 'Spatial Displays as a Means to Increase Pilot Situational Awareness'. In: Spatial Displays and Spatial Instruments, NASA CP10032, pp. 35-1 to 35-12.

4. **Filarsky, S.M. and Hoover, S.W.** (1983). *The command flight path display* Naval Air Development Centre, Warminster, PA.

5. **Grunwald, A.J., Merhav, S.J.** (1978). 'Effectiveness of Basic Display Augmentation in Vehicular Control by Visual Field Cues' *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. SMC-8, No. 9, pp. 679-690.

6. **Grunwald, A.J., Robertson, J.B., and Hatfield, J.J.** (1980). 'Evaluation of a Computer-Generated Perspective Tunnel Display for Flight-Path Following', *NASA TP1736*, Langley, VA.

7. **Grunwald, A.J.** (1984). 'Tunnel Display for Four-Dimensional Fixed-Wing Aircraft Approaches' *Journal of Guidance* Vol. 7 No. 3, pp. 369-377.

8. **Grunwald, A.J.** (1996). 'Improved Tunnel Display for Curved Trajectory Following: Control Considerations'. *Journal of Guidance, Control, and Dynamics*, Vol. 19, No. 2, pp. 370-377.

9. **Kaiser, M.K., and Mowafy, L.** (1993). 'Visual Information for Judging Temporal Range', *Proc. of Piloting Vertical Flight Aircraft*, pp. 4.23-4.27, San Francisco, CA.

10. **Knox, C.E. and Leavitt, J.** (1977). 'Description of Path-in-the-Sky Contact Analog Piloting Display', *NASA TM74057*, Langley, VA.

11. **Owen, D.H.** (1990). 'Perception & control of changes in self-motion: A functional approach to the study of information and skill' In: R. Warren and A.H. Wertheim (Eds.), *The Perception and Control of Self Motion*. Hillsdale, N.J.

12. **Parrish, R.V., Busquets, A.M., Williams, S.P., and Nold, D.E.** (1994). 'Spatial Awareness Comparisons Between Large-Screen, Integrated Pictorial Displays and Conventional EFIS Displays During Simulated Landing Approaches', *NASA Technical Paper 3467*, Langley, VA.

13. **Regal, D. and Whittington, D.** (1995). 'Guidance Symbology for Curved Flight Paths', *Proceedings of the Eight Symposium on Aviation Psychology*, Columbus, OH.

14. **Reising, J., Barthelemy, K. and Hartsock, D.** (1989). 'Pathway-In-The-Sky Evaluation', *Proceedings of the Fifth Symposium on Aviation Psychology*, Columbus, OH.

15. **Theunissen, E.** (1993). 'A Primary Flight Display for Four-Dimensional Guidance and Navigation - Influence of Tunnel Size and Level of Additional Information on Pilot Performance and Control Behaviour', *Proceedings of the AIAA Flight Simulation Technologies Conference*, pp. 140-146, Monterey, CA.

16. **Theunissen, E.** (1994). 'Factors Influencing the Design of Perspective Flightpath Displays for Guidance and Navigation', *Displays*, Vol. 15, No. 4, pp. 241-254.

17. **Theunissen, E. and Mulder, M.** (1995). 'Error-Neglecting Control with Perspective Flightpath Displays', *Proceedings of the Eight International Symposium on Aviation Psychology*, Columbus, OH.

18. **Theunissen, E. and Mulder, M.** (1995). 'Availability and Use of Information in Perspective Flightpath Displays' *Proceedings of the AIAA Flight Simulation Technologies Conference*, pp. 137-147, August 7-9, Baltimore, MD.

19. **Wattler Jr., J.F. and Mulley, W.G.** (1977). 'Feasibility Demonstration of the Earth-Referenced Maneuvering Flight Path Display'. Presented at the AIAA Guidance and Control Conference, Hollywood, FL.

20. **Way, T.C., Hornsby, M.E., Gilmour, J.D., Edwards, R.E. and Hobbs, R.E.** (1984). 'Pictorial Format Display Evaluation', *AFWAL-TR-84-3036*, Wright Patterson AFB, OH.

21. **Wickens, C.D.** (1984). *Engineering Psychology and Human Performance*, Charles E. Merrill Publishing Company, Columbus, OH.

22. **Wickens, C.D., Haskell, I., and Harte, K.** (1989b). 'Perspective Flight Path Displays', *Final Technical Report ARL-89-2/Boeing-89-1*.ARL, Institute of Aviation, University of Illinois at Urbana-Champaign.

AIAA-96-3553-CP

MULTI-AGENT FLIGHT SIMULATION WITH ROBUST SITUATION GENERATION

Eric N. Johnson[*]
The Charles Stark Draper Laboratory, Inc.
Cambridge, Massachusetts USA

R. John Hansman[†]
Aeronautical Systems Laboratory
Department of Aeronautics and Astronautics
Massachusetts Institute of Technology
Cambridge, Massachusetts USA

Abstract

Air traffic management systems, on-board aircraft systems, and proposed airspace management structures such as free-flight are often operated with human operators in a simulated environment. This is necessary for effective design and testing of these systems, as well as the training of the operators. For these experiments or training scenarios, success often depends upon human subjects experiencing one-or-more specific situations. Reliably generating these specific situations is often difficult because subjects may not act consistently or as expected; and these variations can affect the system. Generating specific situations in the presence of this uncertainty is referred to as robust situation generation, robust because the situations must occur within a range of possible actions by the subject. A robust situation generation architecture was developed to support flight simulation tests of air transport cockpit systems. Pseudo-aircraft maneuver within reasonable performance constraints, interact in a realistic manner, and make pre-recorded voice radio communications. The achieved robustness of this system to typical variations in the subject's flight path was explored. It was found to successfully generate specific situations within the performance limitations of the subject-aircraft, pseudo-aircraft, and the script used.

Introduction

An agent is defined here as a component of a system that has a relatively weak coupling with the remainder of the system; such as an aircraft, a helicopter, or an air traffic controller being elements in a simulated air transportation system. A multi-agent simulation mimics an actual system with at least two agents. Agents that a human subject controls are referred to as subject-agents. All other agents are pseudo-agents.

A situation consists of the trajectories and/or actions of one or more of the pseudo-agents over some period of time. The experimental situation may not involve all the pseudo-agents in the simulation. Also, the desired situation often does not fully constrain the trajectory of any particular pseudo-agent. In this context, a desired situation may involve anything from a single pseudo-agent doing a minor action to the complete state make up of every agent in the simulation for a period of time.

If pseudo-agents act autonomously, each with their own goals, knowledge, and processing, then the sequence of interactions between pseudo-agents and the human subject is sensitive to the initial conditions of the simulation and actions by the subject. For example, if a subject-aircraft flies five knots slower than expected over 90 minutes, then the subject-aircraft would be eight NM away from where it was expected to be. Had the desired situation been a collision hazard, a pseudo-aircraft intended to have a near miss might now safely pass as far as 8 NM away. Figure 1 shows both the desired situation (a collision hazard) and the resulting situation if the subject flies slower than expected.

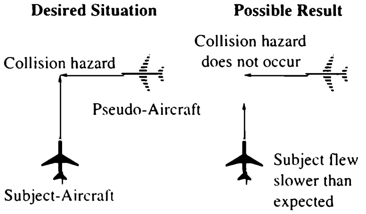


Figure 1 - Sensitivity to Subject Action

Tolerance to actions of the subject-vehicle implies the need for some form of feedback from the subject-vehicle. In current air traffic research, this is typically achieved by having the pseudo-aircraft 'flown' by another human[1]. Another approach is to have an experimenter or instructor change the flight plan of the pseudo-aircraft or subject-aircraft in real time to create

[*] Member AIAA
[†] Professor, Associate Fellow AIAA

specific situations, acting as an air-traffic controller. These options can be quite labor-intensive, and are inherently prone to inconsistent situations; which can confound experiments.

Given the power of computers used for real-time simulation, it has become possible to automate the pseudo-agents and generate specific situations using state feedback from the subject-vehicle. This approach is referred to as robust situation generation. The experimenter or instructor can, within limits, predetermine the situations that the subject is exposed to. These situations can then be replicated for multiple subjects.

In order for robust situation generation to be effective, pseudo-agents must also appear to maneuver realistically from the subject's point of view. The pseudo-agents must individually maneuver within performance constraints. Also, the pseudo-agents must interact with each other properly, in ways that would be normal in the actual system. In many cases, care must be taken only when the subject can perceive the pseudo-agents. These limits on the ability of the pseudo-agents to maneuver limits how robust the system can be.

Figure 2 illustrates how a situation generation system fits in to a multi-agent flight simulation. A subject-aircraft as a single subject-agent, and pseudo-aircraft and controllers are the pseudo-agents. The robust situation generator uses system state feedback to generate scripted situations by providing commands to the pseudo-agents in the form of desired trajectories and event plans, a process described in the next section.

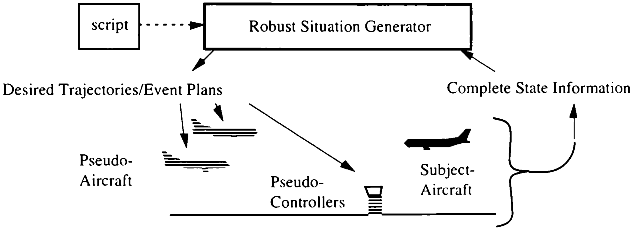

script
Robust Situation Generator
Desired Trajectories/Event Plans
Complete State Information
Pseudo-Aircraft
Pseudo-Controllers
Subject-Aircraft

Figure 2 - Robust Situation Generation System

Approach

To generate specific situations, the robust situation generation architecture utilizes control over where the pseudo-agents go and what they do. The desired trajectory tells pseudo-agents where to go. An event plan tells them what to do and when.

The pseudo-agent model maneuvers along its desired trajectory, resulting in a real-time actual trajectory. Also, it executes an event plan. The event plan contains a list of actions at a criterion to cue each action. This relationship is shown in Figure 3 If the pseudo-agent is a vehicle, this model contains guidance and equations of motion for the vehicle. Otherwise, only the event plan is required.

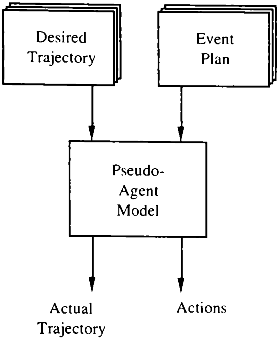


Figure 3 - Pseudo-Agent Model

Pseudo-agent actions need to be realistic when the subject can perceive them. Maintaining a model for each pseudo agent ensures that these agents individually behave in a realistic manner. By using processing that follows the desired trajectory as closely as possible within performance limits, realistic maneuvering of individual pseudo-agents is assured.

Desired Trajectories

Each pseudo-vehicle has a desired trajectory. This desired trajectory can be defined by a list of waypoints. There are many possible desired trajectories, or waypoint types, applicable to robust situation generation. Examples include: traditional position/time waypoints to 'maintain heading, use pitch and speed to collide with subject' or 'stay one mile behind subject'.

A fundamental type of waypoint applicable to robust situation generation is the 4-dimensional waypoint, referred to here as a 4D waypoint. A 4D waypoint is a position in space plus a desired time to be at that point. A series of 4D waypoints define a desired trajectory in space.

One way to use real time subject state feedback is to utilize waypoints that are defined to be relative to the subject-vehicle. A spatially relative waypoint specifies that the pseudo-aircraft is to be in a position relative to the subject-vehicle at a prescribed time. The use of two such waypoints is illustrated in Figure 4, where the pseudo-agent maneuvers to arrive on a parallel course with the subject-vehicle. In this case, the waypoints specify to be 5 NM South of the subject-vehicle at two moments in time.

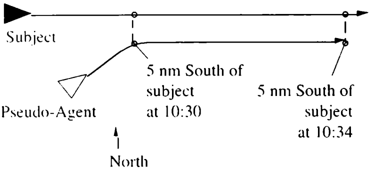

Subject
Pseudo-Agent
5 nm South of subject at 10:30
5 nm South of subject at 10:34
North

Figure 4 - Spatially Subject Relative 4D Waypoints

Depending on the type of multi-agent simulation experiment, other types of subject-relative waypoints may be useful. For example, rather than having the spatial elements relative to the subject-vehicle, it may be useful for the time element to be relative to the subject-vehicle. A waypoint could use normal spatial points, but use a time that is adjusted so the pseudo-agent maintains a prescribed distance from the subject-vehicle. Figure 5 illustrates such a waypoint. The first waypoint is a conventional 4D waypoint with a normal desired time. The second waypoint is a prescribed position with a desired time that is adjusted continuously so the pseudo-agent maintains a prescribed distance from the subject-vehicle. This waypoint uses speed changes to accomplish the specified range between the two vehicles.

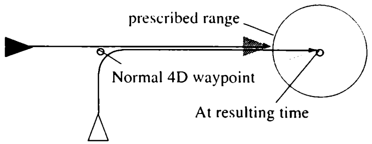


Figure 5 - Subject Range Relative 4D Waypoint

Clearly, many other possibilities for subject relative 4D waypoints exist. Any particular simulation may require one or more type of 4D subject relative waypoint to accomplish its goals.

Event Plan

Pseudo-agents need to do more than just maneuver relative to the subject-vehicle. Pseudo-agents that do radio communications, configuration changes, etc. may need to be a part of a simulation. Anything a pseudo-agent does beyond movement is referred to in this work as an event.

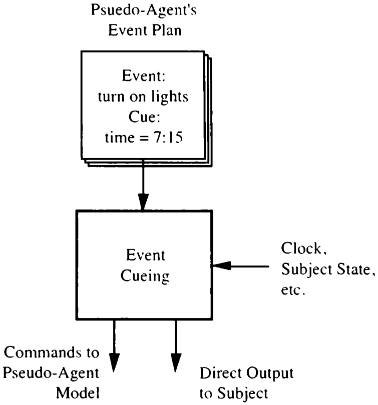

Psuedo-Agent's Event Plan
Event: turn on lights Cue: time = 7:15
Event Cueing
Clock, Subject State, etc.
Commands to Pseudo-Agent Model
Direct Output to Subject

Figure 6 - Event Cueing, Single Pseudo-Agent

Events are cued by some criteria; such as time, subject-vehicle ETA to a key point, or subject-vehicle location. An event plan is assigned and maintained for each pseudo-agent, Figure 6. From this plan, events are cued and result in discrete actions by that pseudo-agent. By allowing events to be cued on criteria other than time, increased robustness to varied subject actions is obtained.

As illustrated in Figure 6, events can result in either direct output to the subject or commands to the pseudo-agent model or perhaps both. An example of the former would be a radio transmission on a frequency the subject is listening to. Examples of the latter are firing a cannon, lowering landing gear, or turning off lights.

Amendments

From time to time, the situation generation controller or human experimenter may want to make discrete changes to the desired trajectories and event plans of one or more of the pseudo-agents. In terms of the architecture presented, an amendment contains waypoints and events for one or more of the pseudo-agents in the simulation. The amendment also has a cueing criterion.

In general, an amendment cueing criterion is a logical expression. For example, an amendment could be scripted to occur when time is greater than 45 seconds, subject speed is less than 200 knots, or some distance is less than 5 nautical miles. The use of other amendment cues can be used to increase robustness.

There are a number of different ways to use amendments and amendment cues to facilitate robust situation generation. Some key examples are illustrated in Figure 7. As the subject-vehicle travels through space, its trajectory will be different than the expected subject trajectory. Amendment can be used to adjust the desired trajectories and event plans of all pseudo-agents based on the arrival time of the subject to an area of space, as shown for the first amendment in Figure 7.

A second example of an amendment is the collision hazard situation also shown in Figure 7. This amendment changes the desired trajectory of one of the pseudo-agents so it maneuvers relative to the subject-vehicle. By use of the amendment, this situation will only begin once the subject is in the proper position. If the subject is late or early, this is accounted for.

A third example of an amendment in Figure 7 is a blunder made by the subject. If the experimenter wants to allow the subject to take different discrete paths, or wants to design for different blunders the subject could make, then amendments could be made to account for these discrete variations.

A very flexible amendment cue is a manual one. This can simply be the experimenter hitting a button to trigger an amendment. Other aspects of the subject or pseudo-agent state, such as ETA, range, velocity, or others are also candidates. Amendment cues can also be logical operations of multiple cues.

Overall Configuration

Two fundamentally different uses are made of subject state feedback in this work. First, amendments and events can be cued based on subject state. Event cues allow pseudo-agent actions to occur at the proper time even when the proper time depends on what the subject does. Amendment cues are used to make changes to pseudo-agent trajectories based on subject state. A second, fundamentally different, use of subject

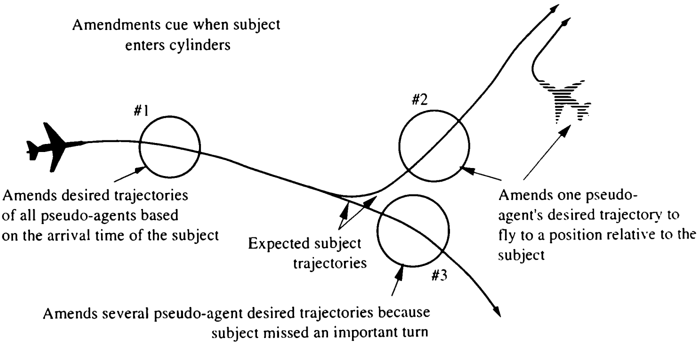

Amendments cue when subject enters cylinders
#1
#2
#3
Amends desired trajectories of all pseudo-agents based on the arrival time of the subject
Expected subject trajectories
Amends one pseudo-agent's desired trajectory to fly to a position relative to the subject
Amends several pseudo-agent desired trajectories because subject missed an important turn

Figure 7 - Amendment Cueing Illustration

state feedback is real time adjustment of a pseudo-agent's desired trajectory by use of subject relative waypoints.

The final robust situation generation architecture must include amendments and amendment cueing, and is depicted in Figure 8. Each amendment contains a desired trajectory update and/or event plan updates for one or more of the pseudo-agents. It also contains a cueing criterion. This cueing criterion can be based on system state or can be triggered by an experimenter. Also, cued amendments can adjust several interacting pseudo-agents simultaneously.

Simulator Setup

The simulation experiments were to be conducted on the Aeronautical Systems Laboratory (ASL) Advanced Cockpit Simulator (ACS). The simulator is centered around a graphics workstation, used to integrate the subject aircraft's dynamics and provide the desired displays. The simulator also provides a Control Display Unit (CDU), Mode Control Panel (MCP), sidestick, and throttle quadrant, shown in Figure 9.

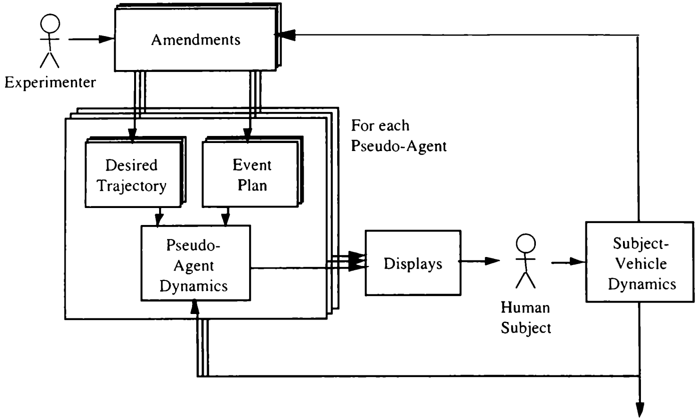


Figure 8 - Overall Configuration

Implementation

This section describes a specific implementation of the robust situation generation architecture. The facility is a tool used for air transportation systems research.

For research involving collision avoidance systems, such as the Terminal alert and Collision Avoidance System (TCAS), and other air transportation concepts such as free-flight, many different types of pseudo-aircraft traffic situations are needed. These include collision and near collision situations, collision hazard situations, and other traffic related situations. Other traffic related situations include expected sequencing around weather, expected holding, etc. to test pilot situation awareness.

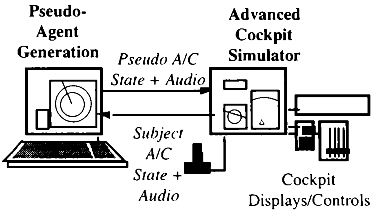


Figure 9 - Simulator Setup

Pseudo-aircraft were generated on a machine separate from the cockpit simulator, also shown in Figure 9. This created an experimenter's station that could be placed away from the pilot's display, in another room if desired. A display of all aircraft in the simulation was developed for use in writing the scripts and monitoring progress during an experiment. The result is the experimenter's display, shown in Figure 10.

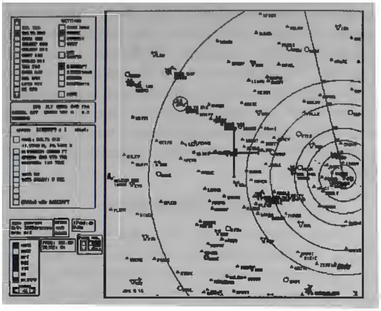


Figure 10 - Experimenter's Display

The experimenter's display is essentially an electronic map of an area. All aircraft, including the subject-aircraft, are shown as symbols at their proper locations. Airports, navigation fixes, intersections, and radio navigation aids are shown. In a separate window on the screen, a variety of information about the subject-aircraft is displayed. This information is received from the ACS and is shown as a reference for the experimenter. Information about pseudo-aircraft can be displayed, including the waypoints the pseudo-aircraft is flying through

Pseudo-Aircraft Modeling

Multi-agent flight simulation relies on realistic modeling of the pseudo-aircraft. This includes equations of motion, performance limitations, and a guidance model. The specifications of planned experiments directly determine the fidelity and accuracy required of this model.

The requirements for a TCAS display imply that pseudo-aircraft states must result in realistic update of latitude, longitude, and altitude. Many proposed enhanced traffic displays, as well as useful TCAS alerts, imply that airspeed, vertical speed or Flight Path Angle (FPA), and heading should also be outputs. Because these experiments do not require any information about pseudo-aircraft conventional control locations (aileron, elevator, rudder), a rather simple aircraft model can be used.

Different aircraft types are modeled through database of aircraft performance parameters. The parameters are then used in a generic performance limits structure for all pseudo-aircraft, summarized in Table 1.

	lower bound	upper bound
Ground speed	stall speed & bottom of power curve based on current environment	regulations & cruise mach number based on current environment
Flight path angle	best attainable in steady state at current altitude within speed limits	
Roll angle	arbitrary	
Ground speed rate	best available given altitude, airspeed, and flight path angle	
Flight path angle rate	lift coefficient & structural load factor	
Roll rate	best available at current airspeed	

Table 1 - Performance Limitations Summary

The performance limitations, in general, need only be active for a particular pseudo-aircraft when that vehicle can be perceived by the subject. This has the potential to improve situation robustness. For this implementation the benefit is small due to the range of the TCAS traffic display.

Voice Generation

Pseudo-aircraft PLI is accomplished by organizing individual pseudo-aircraft voice radio transmissions as events. This is done by digitally recording them ahead of time, and then using the robust situation generation architecture to play them back at the proper time.

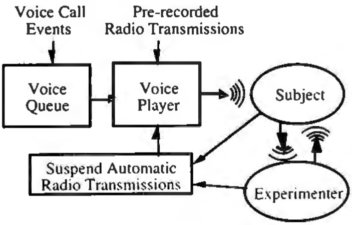

Voice Call Events
Pre-recorded Radio Transmissions
Voice Queue
Voice Player
Subject
Suspend Automatic Radio Transmissions
Experimenter

Figure 11 - Voice Generation

Once a radio transmission event is cued, it goes into a voice queue. It is used in combination with the voice player, Figure 11, to prevent more than one call from occurring at the same time, to suspend voice calls when a human is transmitting, and to ensure that the subject hears only those transmissions on the frequency selected. Included in the definition of each voice call is a frequency that it is transmitted on, a priority value, a maximum wait time, and the identification of the digital recording to play back.

The priority value prevents important transmissions from having to wait too long. Transmissions with higher priority simply skip those of lower priority when entering the queue. A maximum wait time is used to delete a radio communication event when it is no longer relevant, to prevent clearances for unimportant aircraft from occurring after the time period during which they make sense.

An experimenter acts as the controller that the subject is currently communicating with. Although most of the controller's voice calls can be scripted and pre-recorded, all of the possible requests of the subject cannot be realistically prepared for.

Transmissions by the controllers to other aircraft are normally pre-recorded and scripted as events. These voice calls are normally tied to pseudo-aircraft transmissions such that one is played immediately after the other. For example, the sound recording:

KBOS Tower: *United 111, you are cleared to land, runway 4 left.*

would be immediately followed by:

United 111: *United 111, cleared to land 4 left.*

without interruption. This is accomplished simply by having the second transmission cue be the execution of the first. The only way they can be split up is if a higher priority message enters the cue while the first one is playing, as desired.

The subject has a communications radio control console where the transmitting/receiving communications frequency and volume can be changed. The voice queue is suspended when the subject or experimenter-controller transmits, and restarted manually by the experimenter. At this point, radio transmission events would resume playback.

Script Development

To script the flight of numerous aircraft over a significant length of time is not a trivial task. Add to this the creation of specific situations for a subject with varied actions, and it is clear that a critical aspect of this approach to multi-agent simulation is writing the script for the experiment.

For this implementation, an effective way to write and edit scripts was to include tools specifically for this purpose in the experimenter's station, Figure 10. A list of all aircraft is shown at the left of the screen, allowing the user to select specific aircraft. Four other menus can be brought up to modify the robust situation generation script: aircraft, amendment, event, and waypoint. An organized way to record the numerous digital audio recordings to be played back is a necessary component of this type of system. Due to the large number of calls from any individual, it was effective to make an interactive program for the specific purpose of recording.

Achieved Robustness

A test script, or flight, was used to evaluate the robust situation generation architecture and the implementation developed. The expected flight path of test script subject-aircraft is depicted in Figure 12. The subject-aircraft starts at 23,000 feet above LVZ (Wilkes Barre) VOR and proceeds to a landing at New York's JFK airport runway 31 Left. The expected flight path is defined by a series of 4D waypoints. Each point has a latitude, longitude, altitude, and time. The subject receives clearances as necessary to match the expected flight path as closely as possible. The robust situation generation architecture causes specific situations to happen even when the subject varies from the expected path or speed. The general locations of the three situations are labeled on the figure as A, B, and C.

There are three situations included in the test script. First, the subject is to see and hear aircraft ahead request lower altitudes due to turbulence at 19,000 feet. This requires several pseudo-aircraft to fly similar flight paths as the subject while maintaining a scripted separation from the subject and each other. It also requires radio communications from the pseudo-aircraft. It is scripted to occur in area A shown on Figure 12.

The second situation is a TCAS Traffic Advisory (TA) caused by an aircraft passing below. This is to appear normal to the subject. A pseudo-aircraft is to pass 2000 feet below the subject on a perpendicular course while the subject is in area B in Figure 12.

The final situation is a collision hazard while on final approach to runway 31L at JFK, shown as area C in Figure 12. The intruder pseudo-aircraft flies to a scripted location relative to the subject-aircraft on its parallel approach to runway 31R. It deviates from its flight path and creates a collision hazard at five NM from touchdown. The intruder and subject each get a

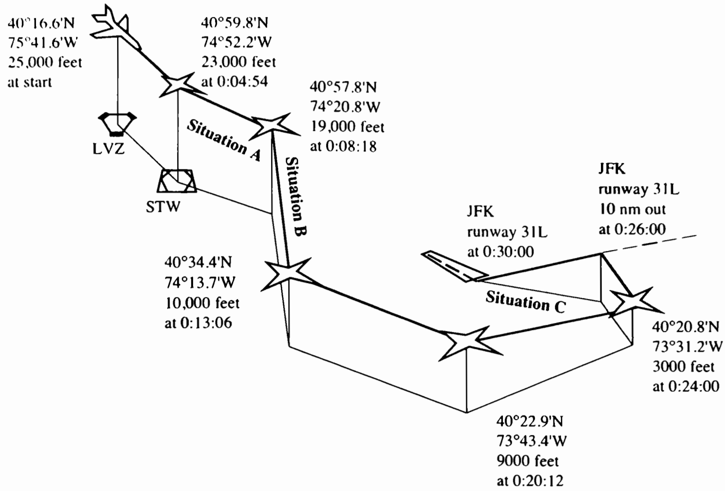


Figure 12 - Nominal Subject Flight Path

TCAS Resolution Advisory (RA) that will generate avoidance commands for both aircraft.

Reasonable background traffic, in the form of pseudo-aircraft not directly involved in any of the above situations, were also used. All aircraft, including this background traffic, must perform within reasonable performance limits and interact properly for an individual flight to be a success.

The first step in writing the test script was determining a nominal flight path for the subject-aircraft, shown in Figure 12. If branches of significantly different flight paths are to be allowed, these other branches should also be determined. In the case of the test script, the only branch allowed occurs during situation A. If the subject requests a lower altitude, it is given. If not, the new altitude will be given at the end of situation A.

The 4D waypoints for the pseudo-aircraft are then created based on the subject-aircraft expected flight path. This approach ensures that the highest fidelity pseudo-aircraft generation occurs were it is needed, in view of the subject. In this case, standard arrival and departure waypoint sets are stored for JFK, Newark, and La Guardia runways. These sets were used repeatedly to eventually define the flight paths of 34 pseudo-aircraft.

The next step taken in this demonstration flight was to split the flight into 15 amendments. The first 14 amendments are designed to cue approximately every two minutes. They provide waypoint updates for all active pseudo-aircraft. The amendment cue for each will be subject-aircraft ETA of less than one minute to a point on the map. By using this approach; if the subject travels slower or faster than expected, amendments will be cued later or earlier respectively. This effectively adjusts the pseudo-aircraft waypoint times every two minutes to variation in subject-aircraft speed.

The complete test script amendment list is shown in Table 2. The flight contains three situations that are critical to the experiment. The turbulence reports situation corresponds to the third amendment in the list. The TA corresponds to the fifth. The parallel approach RA begins when the 14th amendment is cued, and an additional amendment is cued so that the RA occurs at 5 NM from the runway threshold. The RA amendment updates only the pseudo-aircraft that will cause the RA. The other amendments are necessary to maneuver the pseudo-aircraft realistically and place them for the three experiment critical situations in a robust manner.

Amendment	Expected cue time	note:
1	0:00:00	Active at start-up (initial set of waypoints and events)
2	0:02:00	
3	0:04:00	Turbulence reported at 19,000 feet from pseudo-aircraft situated ahead of the subject-aircraft
4	0:06:00	
5	0:08:00	TA situation; TA aircraft gets a subject relative waypoint 2000 feet directly below the subject-aircraft
6	0:10:00	
7	0:12:00	
8	0:14:00	
9	0:16:00	
10	0:18:00	
11	0:20:00	
12	0:22:00	
13	0:24:00	
14	0:26:00	Intruder aircraft gets subject relative waypoints on the parallel approach in order to get in the proper position
RA	0:26:45	Collision hazard (RA); intruder flies directly into the subject-aircraft until RA occurs

Table 2 - Test Script Amendments

Achieved robustness was evaluated by varying the subject's flight path to extremes of speed and lateral position error, as well as subject blunder errors. They were varied to the point that the test script and the robust situation generation architecture could no longer adequately control the pseudo-agents, generate desired situations, or when the extreme of the subject-aircraft's performance envelope has been reached. Speed and lateral position errors are differences between the expected subject-aircraft flight path and the actual flight path. Blunder errors were wrong turns and incorrect altitude commands made by the subject.

Speed Error

The achieved robustness to speed variation was explored by varying the speed of the subject by a multiplicative factor. The system was tested by flying the subject-aircraft at 120, 110, 90, and 80% of the expected speed profile, which contained speed near both the upper and lower bounds of the subject-aircraft's speed envelope.

The resulting amendment cue times are shown in Figure 13. Early in the 110 and 120% cases the subject-aircraft was performance limited. This is shown by the curves lying roughly on top of each other in the first 5 amendments. Though not apparent from the figure, the subject-aircraft flew unrealistically close to stall speed for much of the later part of the slowest test, 80% speed.

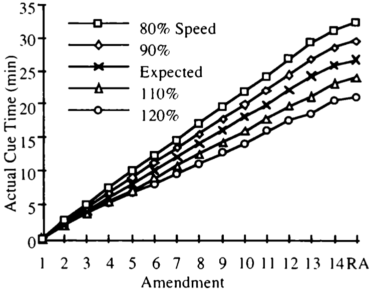


Figure 13 - Actual Cue Times for Different Subject Speeds

For these tests, the three experiment critical situations and the background traffic were observed. For all but the fastest test, 120% speed, experiment critical situations occurred as scripted and background traffic appeared to maneuver properly.

For 120% case, it was observed that some background traffic could not keep up with the experiment due to the 250 knots speed limit below 10,000 feet. As a result, some turns were cut short and trailing distance in some landing sequences became unreasonably small. For the same reason, the pseudo-aircraft involved in the parallel approach RA could not arrive in time to cause a collision hazard at 5 NM from the runway as scripted. A summary of these results is shown in Table 3.

	Turbulence			Back-
Speed	Reports	TA	RA	ground
80%	yes	yes	yes	yes
90	yes	yes	yes	yes
110	yes	yes	yes	yes
120	yes	yes	no	no

Table 3 - Results of Subject Speed Variation

Although the subject is highly unlikely to fly the 120% trajectory, it is possible that the waypoints given to offending pseudo-aircraft could be modified to allow the system to work under these conditions. The approach would be to have these aircraft fly slower when the subject is on the expected flight path. Clearly, there is a balance between tolerance allowed at the bottom and top of the subject's speed range. The experimenter can shift the range of speeds up and down by adjusting pseudo-aircraft waypoints. As shown, this speed range available to the experimenter in this implementation is approximately the same as the subject-aircraft's performance limits.

Position Error

Another way the subject can vary flight path from expected, beyond speed, is to fly slightly off the expected course. This can manifest itself as being slightly left, right, above, or below the expected flight path.

Achieved robustness to position error was tested by flying the subject-aircraft one, two, and four NM right of the expected flight path. These errors are extreme for a transport category aircraft following an ATC clearance, but were chosen in order to test the limitations of this robust situation generation system. No position error was included once the subject was established on the localizer.

Position errors of one and two NM right of the expected course had no effect on the system. This was not the case with a position error of four NM. In this case, using the ETA of one minute amendment cue meant that amendments would not cue if the subject was flying slower than four NM per minute. This corresponds to a ground speed of 240 knots. Once the subject-aircraft speed dropped below this value, which happened shortly after the TA situation, new amendments were not properly cued. This caused the simulation to longer be tolerant to subject actions, so the parallel approach situation and the background traffic were no longer assured. A summary of these results is shown in Table 4.

	Turbulence			Back-
Error	Reports	TA	RA	ground
1 NM	yes	yes	yes	yes
2	yes	yes	yes	yes
4	yes	yes	no	no

Table 4 - Results of Subject Position Error

If an experiment requires tolerance to position errors of this magnitude, four NM, a different amendment cue should be used, perhaps a larger ETA value.

Blunder Error

The final type of subject variation explored is the blunder error. Tolerance to blunder errors was tested by having the subject-aircraft make a key turn at two levels of delay. In all cases the experimenter was assumed to intervene, acting as ATC, and clear the subject to a new heading that will put the subject back on the desired flight path. Achieved robustness was also tested by having the subject-aircraft descend too far when capturing a cleared altitude, and then remain at this lower altitude until it returned to the expected flight path.

The first test was to have the subject aircraft fly 2.5 NM beyond the point at which the base turn was to be initiated, the 9000 feet 0:20:20 waypoint in Figure 12. This could be caused because the subject did not hear the new clearance. This blunder had no effect on the system. When the blunder distance was increased to five NM amendment #12 in Table 2 did not cue. This *might* cause a problem if the subject were to fly at a very different speed than expected during this period of time, which would be unusual for this particular phase of flight, because the pseudo-aircraft had to wait four minutes between trajectory updates rather than the normal two minutes. A summary of these results is shown in Table 5.

	Turbulence			Back-
Blunder Error	Reports	TA	RA	ground
• 2.5 NM late turn to base	yes	yes	yes	yes
• 5 NM late turn to base	yes	yes	yes	see text
• descent to 17,000 rather than 19,000 feet	yes	yes	yes	yes
• descent to 6000 rather than 10,000 feet	yes	yes	yes	yes

Table 5 - Results of Subject Blunder Error

Altitude blunder errors were also tested. The descent to 19,000 feet shown in Figure 12 was lowered to 17,000 feet. In a separate test, the descent to 10,000 feet was increased to 6000 feet. These blunder errors represent extremes of possible mistakes to be made by subjects. Also, these two cases are among the few possible scenarios where the subject-aircraft can descend significantly too far using a reasonable descent rate. In both cases, the system performed as expected, also shown in Table 5.

Summary and Conclusions

A robust situation generation approach has been developed. This approach utilizes subject feedback in two fundamental ways. First, the trajectory of a pseudo-agent can be adjusted continuously in response to the motion of the subject. Second, discrete qualitative amendments to the agents' trajectories can be cued by some aspect of the subject's current state. In addition, discrete actions can be similarly cued for the pseudo-agents, such as turning on lights or lowering landing gear.

The robust situation generation approach was implemented for an air transportation system research facility. Experiments required voice communication to be heard by a human subject from other aircraft. It also required specific types of collision hazards between the subject and other aircraft. A pseudo-aircraft model and other related software was developed to implement the situation generation architecture for use in this type of research.

A test situation generation script, designed to be a part of an air transportation research experiment, was developed. This script contained three situations that were critical to present to the subject. The script included fifteen amendments, where these amendments organized background aircraft and led aircraft involved in the three experiment critical situations to their proper positions. The achieved robustness of this test script to variations in subject actions was explored. This analysis included varying the subject-aircraft speed and position accuracy, as well as testing blunders by the subject, such as missing a turn.

Achieved robustness indicated that the system allows specific situations to be generated for subjects who perform within a reasonably large envelope of possible action. In cases where the system failed, the subject was performing at an extreme of possible action or a limitation was found in the script that could be rectified if needed.

The development of a script is an important element of the system presented. Scripting the flight of multiple aircraft in a crowded sky is not a trivial task even before attempting to generate specific situations. This effort can be reduced using software specifically developed for the purpose of script writing.

Triggering voice communications using the robust situation generation architecture, with the implementation of a voice queue, was found to be a powerful technique. It allows pseudo-aircraft radio transmissions to be pre-recorded. This eliminates the need for a large number of 'pseudo-pilots' that would normally provide these transmissions from manned remote stations.

The architecture developed has proven to be effective for air transportation research. It could also be applied other applications, where the details of the implementation would differ. Any system that must coordinate one or more pseudo-agents to give specific situations to a maneuvering subject-vehicle is a candidate for a robust situation generation scheme.

References

1. Bayne, S., Schwartz K., Smithwick M., and Weske R. A., *Pseudo Aircraft Systems (PAS) Documentation Package*, NASA Ames Research Center, Release 1.1, November 1990

2. Midkiff, A. H. & Hansman, R. J. *Identification of Important "Party Line" Information Elements and the Implications of Situational Awareness in the Datalink Environment*, MIT Aeronautical Systems Laboratory Report ASL-92-2, May 1992

3. *Minimum Operational Performance Standards for Traffic Alert and Collision Avoidance System (TCAS) Airborne Equipment*, Volumes 1 and 2, RTCA/DO-185, September 1983

4. Nolan, M. S. *Fundamentals of Air Traffic Control*, Wadsworth, Belmont California 1994

5. Pritchett, A. R. & Hansman, R. J. *Variations in Party Line Information Requirements for Flight Crew Situation Awareness in the Datalink Environment*, MIT Aeronautical Systems Laboratory Report ASL-94-5, May 1994

6. Simpson, R. W., *Engineering of Air Traffic Control Systems*, Flight Transportation Laboratory, August 1993 Draft copy

7. Stevens, B. L. & Lewis, F. L., *Aircraft Control and Simulation*, Wiley, New York 1992

8. Ward, D. T., *Introduction to Flight Test Engineering*, Elsevier, Amsterdam 1993

AIAA-96-3554-CP

FLIGHT SIMULATOR TESTING OF COCKPIT TRAFFIC DISPLAYS USING ROBUST SITUATION GENERATION

Amy R. Pritchett
Richard Barhydt*
R. John Hansman†
Aeronautical Systems Laboratory
Massachusetts Institute of Technology
Cambridge, MA

Eric N. Johnson*
The Charles Stark Draper Laboratory, Inc.
Cambridge, MA

Abstract

Flight simulator experiments testing cockpit traffic displays are underway, for applications such as Free Flight and closely-spaced parallel approaches. To meet their repeatability requirements, the recently developed Robust Situation Generation (RSG) architecture is being used. RSG uses feedback of the subject aircraft states, such that the pseudo-agents will act to consistently generate the required situations, regardless of variations in the subjects' actions.

However, the use of RSG mandates changes in the experiment design to include the development and testing of the RSG scripts of situations. The RSG system follows pre-set scripts involving several variables; some iteration of the script design may be required to match experiment specifications. To generate any one desired situation, the experiment designer can use one of several possible RSG command sequences. Each of these command sequences have different characteristics and will be discussed. The robustness and fidelity of the command sequences depends strongly on the detail and time-scales of the pre-determined script, and reasonable time-scales for different situations will be given. Finally, RSG allows for greater control over the situations that will occur, allowing for more careful evaluation of the desired nature of the situations. These variations will be discussed.

Introduction and Motivation

New technologies are enabling several new features in air transport aircraft, including cockpit presentation of external conditions such as weather and nearby traffic, collision alert and avoidance systems, and new air traffic control strategies such as Free Flight. The development and verification of these new aircraft systems require flight simulation evaluations which may measure any of several variables, including the overall system performance, flight crew situation awareness, and the characteristics of flight crew reactions to the external conditions.

These measurements can be made by presenting pilots with specific situations to which they must react. These situations must have several qualities. First, situations should be chosen for which standard operational criteria demand a measurable response. For example, a recent simulator experiment gave pilots an enhanced traffic display which allowed them to monitor an aircraft on a parallel approach. Whenever this aircraft deviated towards them, action was required to avert a collision; a lack of action by the pilots could be considered to represent a lack of pilot situation awareness.

Second, these situations should be chosen to cover the domain of important situations in which the system is expected to perform. For example, testing of a final prototype system may include situations which test all conditions given in the system design specifications.

Finally, the situations must represent believable and recognizable occurrences to which the subject can be expected to react as they would in the real, non-simulated environment. For example, an experiment was developed in which the subjects were flying an air transport simulator and believed they were overhearing the voice transmissions from other air transport aircraft. Therefore, the 'Potential Collision'

* Member AIAA
† Professor, Associate Fellow AIAA

situations were staged to happen at a rate which was physically reasonable and were carefully scripted to portray to the subject a believable scenario of pilot confusion and/or mechanical failure on the part of the intruding aircraft.

In order for consistent, significant results to be attained, these situations must occur reliably and in the same manner between different subject-pilots. This requirement is confounded by the requirement that, for reasonable fidelity, the pilots must be able to fly the simulator without artificial constraints. Therefore, any variance in their own actions and trajectory can cause the desired situations to not happen, or to evolve differently for different pilots. To meet these requirements, the recently developed Robust Situation Generation (RSG) architecture is being used.

RSG enables an experimenter to define, at a high level, the situations in pre-determined scripts; during experiment runs RSG uses feedback of any variations of the subject's actions about nominal to modify the behaviour of the pseudo-aircraft and Air Traffic Control calls in order to ensure the desired situations occur with the pre-scripted characteristics. This feedback process is shown in Figure 1.

RSG has been found able to reliably produce situations involving interactions between the subject and other aircraft or Air Traffic Control, without requiring experimenters to act as pseudo-pilots. Meeting the three requirements listed for the situations listed above, it has been used to generate a wide-variety of situations which are believable and require a testable response. In addition, because the simulation flights with RSG use feedback to achieve pre-scripted target situations, the situations happen consistently between pilots, allowing for more accurate analysis and comparison of pilot reactions and system performance.

RSG has also enabled greater accuracy in the situations. For example, the exact miss distance in a simulated near mid-air collision between the subject and a pseudo-aircraft can be specified and attained with greater accuracy that the use of pseudo-pilots alone can achieve. In addition, RSG can allow for more complex situations. This complexity may be increased by its adding more aircraft, by making the required actions of the aircraft and Air Traffic Controllers more intricate and dependent on the subject's actions, and by increasing the subject's ability to observe surrounding traffic through better displays and audio communications.

In developing a flight simulator experiment, the desired situations must be pre-scripted in a set of high-level commands to the aircraft and Air Traffic Controllers simulated by RSG. A simple scenario which steers a pseudo-aircraft, Flight 234, to a specified position relative to the subject is shown in Figure 2.

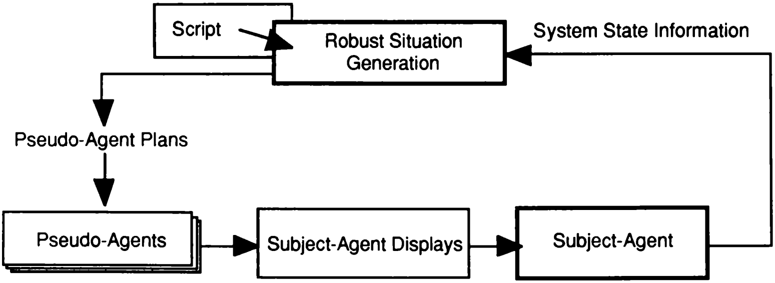


Figure 1. Use of Subject-Aircraft State Feedback to Control Pseudo-Agent Actions

Trajectory Commands
Flight 234:
Arrive at JFK VOR at 2:01 GMT,
Then Be 3 Miles North of the Subject
at 2:04 GMT, Co-Altitude

Event Commands
Flight 234:
Play Voices File at JFK VOR in Sequence:

"Departure, Flight 234 is at JFK VOR at 6000 feet"

"Flight 234, Maintain Heading for Now, Immediate Climb to 8000 feet"

" Roger, Leaving 6 for 8, Flight 234"

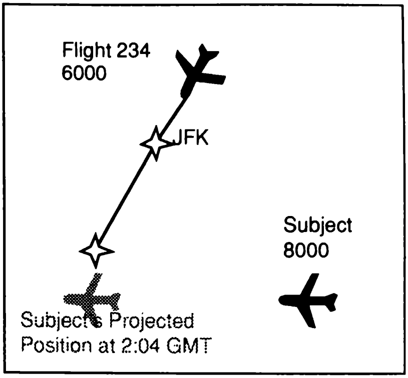


Figure 2. Simple RSG Script Text and Corresponding View on ATC Display

Many high level commands are available for the aircraft, including: steering to four dimension (specified position and time) waypoints, flying to a specified position relative to the subject, cueing of voice communications, and cueing compensatory waypoint sets for aircraft (amendments) when the subject deviates from their nominal flight path.

The large number of RSG commands available for determining the script of required situations causes the scripting process to be under-determined -- several different RSG commands can be used to create the same situation. However, subtle differences in the appearance of situations caused by the RSG commands have been documented through the experience of scripting recent flight simulator experiments.

The detail and complexity of generating the script of required situations has also been found to depend on characteristics of the experiment and of the ability of the subject to observe the aircraft and follow the Air Traffic Control communications. For example, frequent, small amendments must be anticipated during scripting when the subject is expected to deviate often from the nominal flight path or when the subject is given sufficient information about the other aircraft to notice large jumps in a pseudo-aircraft's project flight path.

The final scripts can be very extensive, listing many pseudo-aircraft, each following trajectories which may include a large amount of amendments anticipating subject deviations. Because the text of the scripts can be quite detailed, the implementation of RSG used for recent experiments has included a large ATC-like display with several menus. This interface can be used to script the scenarios interactively by indicating trajectories on the screen and inputting numerical data through the menus. The scenarios can then be tested in fast-time and monitored in real-time during the experiment runs. The RSG screen is shown in Figure 3.

This paper will document the process of generating scripts of desired situations as part of a flight simulator experiment. The variables affecting the choice of different RSG commands and the detail of the script will then be expanded upon; these include the accuracy required of the situation and the ability of the subject to observe the pseudo-agent actions. Recommendations in the use and further development of RSG can then be made.

Experiment Design Using RSG

The use of RSG dramatically changes the process of designing an experiment in several ways. The use of RSG enables more specific detailing of how the situations are to evolve than is otherwise possible, and therefore more exact consideration to the actions of the pseudo-aircraft may be necessary. The development of the script of the process is often iterative as the high-level RSG commands to the pseudo-aircraft are tested against the range of subject deviations from nominal, and should consider the qualities of the desired situation, the ability of the

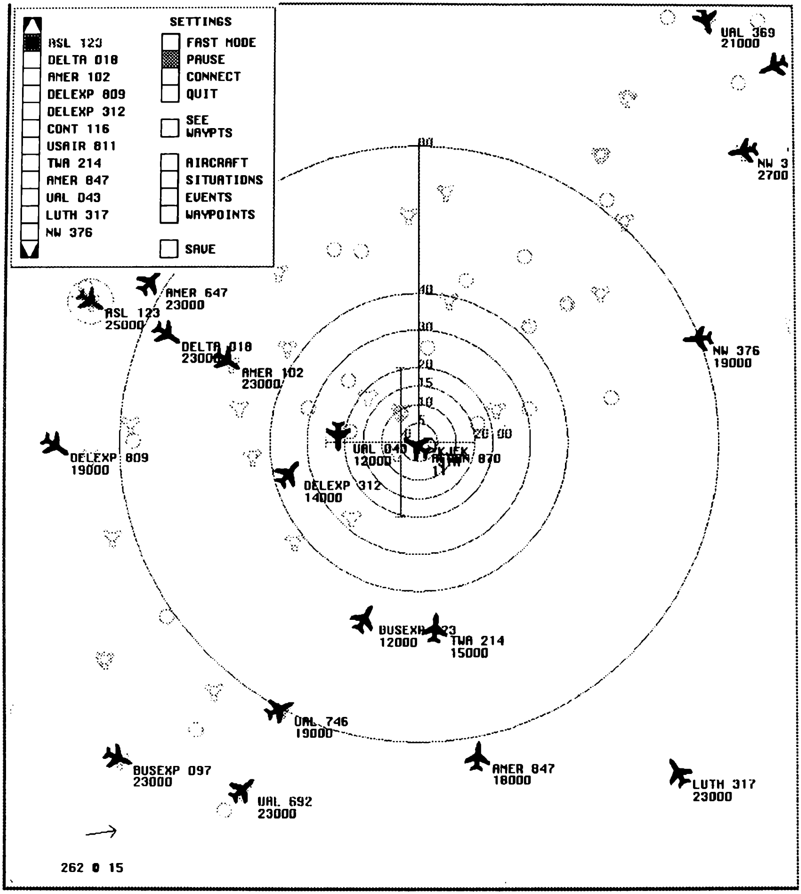


Figure 3. RSG Interface, with Map View of Subject and Pseudo-Aircraft, and Menus for Setting Script Variables

subject to observe the pseudo-agent actions, and the RSG tools available. This section will detail these concepts.

A flow-chart of the experimental design process when using RSG is shown in Figure 4. The first step is to develop a list of situations for the subject to react to which span the range of issues defined by the experimental issues. RSG allows these situations to be defined more precisely than previously possible, such that a 'potential collision' situation can be further characterized, for example, as a having a particular convergence angle or convergence rate in both the lateral and vertical planes; with this ability comes the requirement for the experiment designer to identify the elements of the situation which are to be constrained and those which are free for the RSG architecture to vary during the run to achieve the desired situation.

Using the list of situations, the actions of the aircraft can be calculated in reverse from the most critical situations back to the start of the flight. These actions can be defined in terms of the aircraft trajectory and also in terms of discrete events such as generating their own voice calls to and from Air Traffic Control. These voice calls can be pre-recorded and then played by RSG during the experiment runs.

The exact script of high-level RSG commands to the pseudo-aircraft can then be created. This stage of the experiment design depends the most on an understanding of several factors, as listed in Table 1.

- RSG Commands Available
- Desired Characteristics of Situations
- Desired Types of Measurements
- Subject Ability to Deviate from Nominal
- Subject Observability of Pseudo-Agent Actions

Table 1. Factors in Pre-Determining Script

These factors must be considered in unison to decide upon the resultant characteristics of the script listed in Table 2. The difficulty in developing the RSG scripts lies in the many considerations required in mapping the factors in Table 1 into the script characteristics in Table 2.

RSG Commands Used

Robustness of Script to Subject Deviations

Frequency and Detail of Script Amendments

Table 2. Characteristics of Resultant RSG Script

These factors can be categorized as follows. Several types of RSG commands are available. The trajectory of the pseudo-aircraft can be fixed to absolute earth coordinates or to a variety of subject-relative coordinates, such as steering to subject relative position or tying the pseudo-aircraft speed to the subject's speed.

The RSG commands can also identify events, such as Air Traffic Control voice calls, and turning an aircraft turning its transponder on or off. Other types of discrete events can be added as required by an increased fidelity of the simulation.

The most powerful RSG commands use the ability to generate amendments to the trajectory and events to be executed by any of the pseudo-aircraft. This allows for 'forks' during a flight simulation experiment run; for example, pseudo-aircraft A may be given a new trajectory to follow onto a collision course if the subject turns right around weather, and pseudo-aircraft B is given the collision trajectory if the subject turns left.

Both the events and the amendments are 'cued' during the simulation run based on scripted criteria. These cues can be simple, such as a fixed time or when the subject reaches a fixed ground coordinate, or may be more elaborate. New cueing criteria can also be added to meet the requirements of a situation.

An additional characteristic of the script is the amount of robustness incorporated against variations in the subject's trajectory. This level of robustness can be simple or elaborate. For example, if a 'collision' situation is desired, a simple form of robustness would have a pseudo-aircraft track the subject. A more elaborate scheme would steer several aircraft to tracks near all possible subject trajectories, and would cue an amendment for the pseudo-aircraft nearest to the subject which would command a collision.

The final characteristic of the script is the detail of the script and the frequency with which new trajectories and events are cued through amendments. The best level for this can depend on the time-scale of the situation and on the required accuracy of the situation. For example, compare the task of reliably generating a collision-hazard situation in each of: a closely spaced parallel approach (time to collision is 30 seconds), and an enroute cruise environment (time to collision is several minutes). In the first case, the script must update the intruding pseudo-aircraft's steering commands every couple of seconds in order to achieve the desired situation; in the second, enroute cruise case, steering commands updated every couple of seconds would create an erratic flight path for the pseudo-aircraft whereas steering updates at a lower.

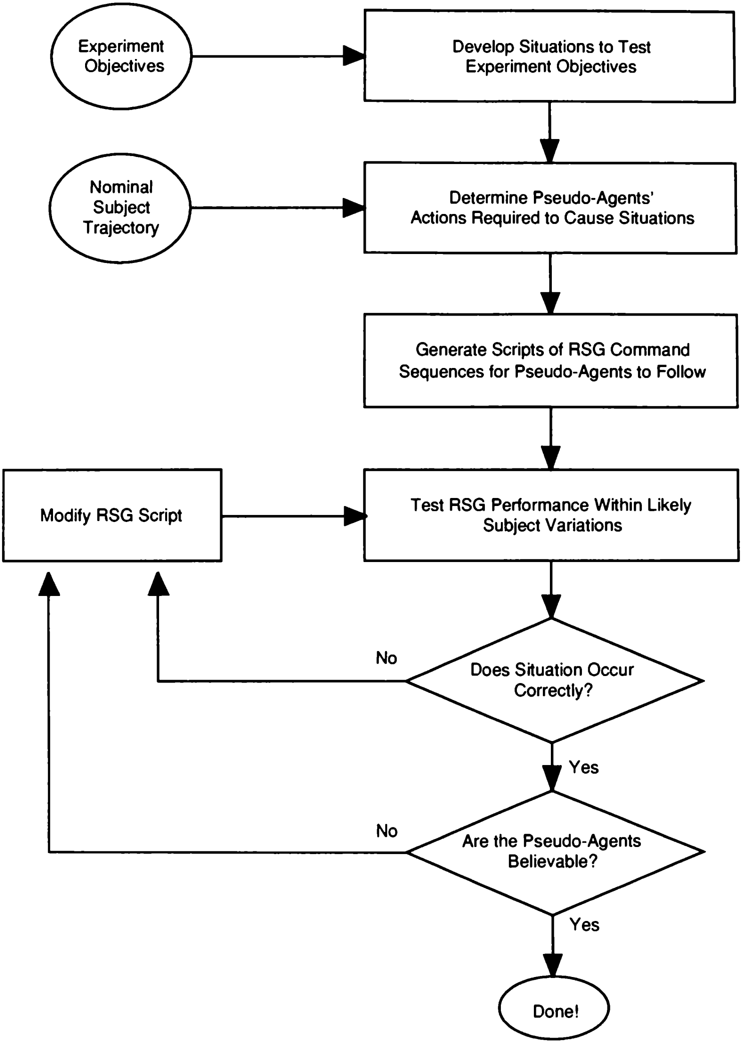

Experiment Objectives
Develop Situations to Test Experiment Objectives
Nominal Subject Trajectory
Determine Pseudo-Agents' Actions Required to Cause Situations
Generate Scripts of RSG Command Sequences for Pseudo-Agents to Follow
Modify RSG Script
Test RSG Performance Within Likely Subject Variations
No
Does Situation Occur Correctly?
Yes
No
Are the Pseudo-Agents Believable?
Yes
Done!

Figure 4. Flow Chart of Experiment Design with RSG

frequency would result in a more believable pseudo-aircraft trajectory and still create the desired situation

The variety of RSG commands, the frequency at which they are updated and the ability to include redundancy against subject deviations gives great flexibility to the experiment designer, but also creates an under-constrained environment. Two factors have a large effect and will be discussed in detail later in this paper: designing the script for situations requiring high accuracy in the pseudo-agent actions and designing the script when subject observability of the pseudo-aircraft is high.

The task of generating the RSG command scripts can be better constrained by more narrowly defining the situations, and by considering the characteristics of the situations which will most strongly be identified by the measurements being used in the experiment. Inclusion of these considerations at the beginning of the scripting process helps promote more exact measurements of the desired factors.

Once a script has been formed, careful testing and re-iteration of the script is usually necessary. The first concern is verifying the desired situations will occur; the second is verifying sufficient experimental fidelity is achieved, ie. that the situations are believable as seen by the subject. This iteration should be carried out for both the nominal trajectory and for any expected subject deviations. Of course, the more subject deviations the simulation is expected to cover, and the higher the required fidelity, the more iterations are to be expected.

To summarize, the process of experimental design has been found to greatly involve the development of the RSG command scripts. RSG enables use of very specific situations with very specific constraints. Therefore, the evolution of the situations throughout the experiment runs can be decided upon early in the experiment design, and tailored to the experiment objectives and measurements.

Many variables are available for commanding the pseudo-aircraft and Air Traffic Control trajectories and actions. Iteration of the script can be continued to produce the required robustness and accuracy.

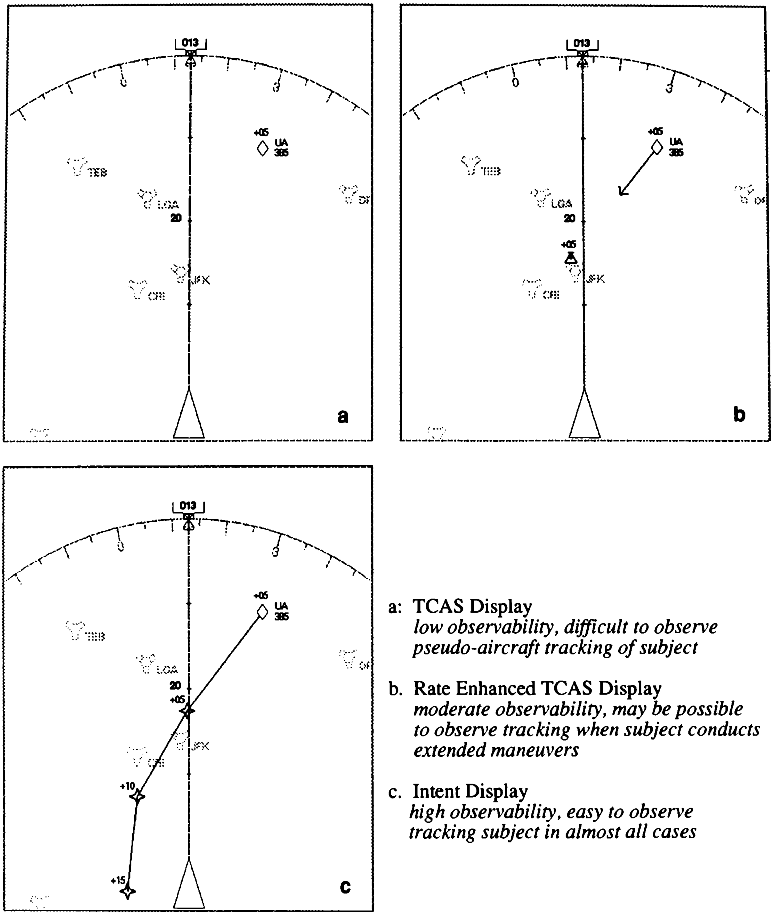
Effects of Subject Observability of Pseudo-Aircraft Actions

As discussed previously, many different RSG commands are available to experimenters when designing RSG scripts. The use of these different commands becomes more constrained as the subject is able to observe more information about intruder aircraft. When a greater amount of traffic information is presented to the subject, realistic situations are more difficult to design and subjects are given more opportunities to observe unrealistic intruder actions. Increased levels of subject observability tend to shift the design process from an under-constrained to an exactly or over-constrained problem. Visual and aural traffic information must cause the subject to believe that he is observing real airplanes flying in a dynamic traffic environment.

When observed by the subject, intruder aircraft must appear to fly realistically. Aircraft performance limitations have been built into the RSG logic to prevent clearly impossible movements. These constraints may make it difficult for pseudo-aircraft to create the desired conflict after a series of unexpected subject maneuvers. A pseudo-aircraft that flies at twice the speed of sound in order to reach a programmed waypoint will not be believable to a subject pilot. One way to sidestep this issue is to keep the pseudo-aircraft not visible to the subject until it is needed to create a conflict. Performance limitations may be relaxed when the intruder is off-screen and unobservable to the subject, therefore allowing it to be placed at any location convenient to the experimenter. Then, when the pseudo-aircraft is on the subject's screens, the performance limitations can be tightened such that the pseudo-aircraft behaviour is believable.

Another aspect of subject observability is the amount of information displayed to the subject about the other aircraft. By showing more information about an intruder's future trajectory, the subject has a better chance to observe unrealistic maneuvers and path changes. This can have a significant impact on the apparent fidelity of the simulation; for example, an intruder's path should not change continuously in response to a maneuver by the subject. These issues may limit the use of subject relative waypoints as more pseudo-aircraft flight path information is shown.

As an example of how these problems can be observed, consider the three traffic displays shown in Figure 5, each showing progressively greater amounts of intent information. Each display shows the same traffic conflict over JFK, caused by an intruder aircraft approaching from the northeast. At the lowest level of observability is the display shown in Figure 5a, one implementation of the traffic display installed with current Traffic alert and Collision Avoidance System (TCAS). Since this display only shows an aircraft's current position, the subject is unaware of any future intentions of the intruder. This gives the experimenter more liberty in designing the script. A subject is unlikely to suspect that he is being tracked when he is



a: TCAS Display
low observability, difficult to observe pseudo-aircraft tracking of subject

b. Rate Enhanced TCAS Display
moderate observability, may be possible to observe tracking when subject conducts extended maneuvers

c. Intent Display
high observability, easy to observe tracking subject in almost all cases

Figure 5
Effects of Showing Aircraft Intent Information on Ability of Subject to Perceive Being Tracked by Pseudo-Aircraft

only able to observe the intruder's current position. As more information is added to the traffic display, the experimenter must take more care in scripting the intruder's trajectory.

The Rate Enhanced TCAS Display, shown in Figure 5b predicts an intruder's future position based on its current state. This display, showing the intruder's location and altitude relative to the subject at the closest point of approach, makes it easier for the subject to more closely follow the intruder's actions. A simple RSG design consisting of a single amendment and subject relative waypoints would cause the closest point of approach symbol to continuously maintain the same orientation with respect to the subject. Since the intruder trajectory is only shown at one point, however, the subject may not have enough time during a small course change to recognize intruder tracking. When using this display, possible subject actions should be carefully considered before settling on a simple RSG script.

The highest case of observability occurs when the Intent Display of Figure 5c is used. The intruder's complete trajectory is presented to the subject and also commands a significant portion of display space. When subject relative waypoints are used, the intruder's entire trajectory is continuously updated during a subject maneuver. This update would almost certainly be noticed by the subject and would contra-indicate the fixed-in-space trajectory the Intent Display is supposed to be providing. Use of a more complicated RSG script is probably mandatory when using this display.

Adding more information to the subject's traffic display requires the experimenter to be more careful in presenting the intruder aircraft's track. Often, this requires the development of a more complicated script, in order to hide unrealistic intruder actions. In some instances, an experimenter may consider reducing the role of subject relative waypoints, or delaying their use until just before the desired conflict. There may also be a need for a large number of amendments, each accommodating a possible subject deviation from the expected flight path.

In addition to displays, voice commands also allow the subject to observe more information about the traffic situation. As with visual information, subject observability increases as more complete information is presented about other aircraft. Increased levels of subject observability require a large number of available voice commands, each providing a high level of detail. The selection of the proper command depends on actions taken by the subject. For instance, in order for the voice communications to match a developing collision trajectory, ATC calls may be required commanding a pseudo-aircraft to climb or descend, depending on whether the subject is higher or lower.

In addition, the timing of an amendment cue could affect the content of the ATC voice communications. If an amendment is cued later than nominally expected, a pseudo-aircraft may need to proceed rapidly to a new point. Words such as 'without delay' or 'immediately' can be used to validate sudden changes in the pseudo-aircraft's trajectory. As the level of detail given by voice communications increases, they must encompass a wider range of possible subject actions. The danger of an incorrect or unrealistic communication being issued increases with greater script complexity.

Providing the subject with more information about the traffic situation requires the experimenter to place greater emphasis on the development and testing of RSG scripts. Detailed displays and voice communications add constraints to the problem, requiring a more complicated script design. A thorough investigation into possible subject actions and the careful use of amendments, waypoints, and events will lead to a realistic and useful model of a traffic situation.

Creating Highly Accurate Situations

This section will deal with the some specific insights into achieving a high accuracy in the steering of the pseudo-aircraft. Depending on the situation, the pseudo-aircraft may need to be at highly defined positions. For example, a recent study of collision avoidance systems for closely spaced parallel approaches required the pseudo-aircraft to arrive within 500 feet of the subject aircraft, despite the variations between subjects in configuring the aircraft for approach and reducing aircraft speed to landing speed; this highly precise situation required more careful scripting than another experiment's enroute situations which only required pseudo-aircraft to fly approximately five miles in front of the subject.

Two elements of the script require careful consideration to achieve highly precise situations: the type of RSG command used, and the time at which they are initiated.

RSG provides two means of commanding pseudo-aircraft trajectories. The first, subject relative waypoints, causes the pseudo-aircraft to be continually steered towards the subject. Subject relative waypoints can be highly accurate in providing an exact position relative to the subject. However, if the subject is constantly modifying their own path the pseudo-aircraft will be constantly adjusting its target position and will therefore have an erratic flight path. In addition, a subject relative waypoint must be attainable by the pseudo-aircraft within its performance limitations.

The second method of commanding pseudo-agent trajectories is the ability to use amendments to give the pseudo-agent a completely new set waypoints to follow. These amendments are cued based on criteria given in the script; the criteria can be simple, such as a fixed time, or can be more intricate, such as when the pseudo-agent is in the proper position to start tracking the subject. When implementing only fixed-space waypoints, however, amendments can be inflexible and do not have the inherent flexibility to account for small changes in the subject's trajectory.

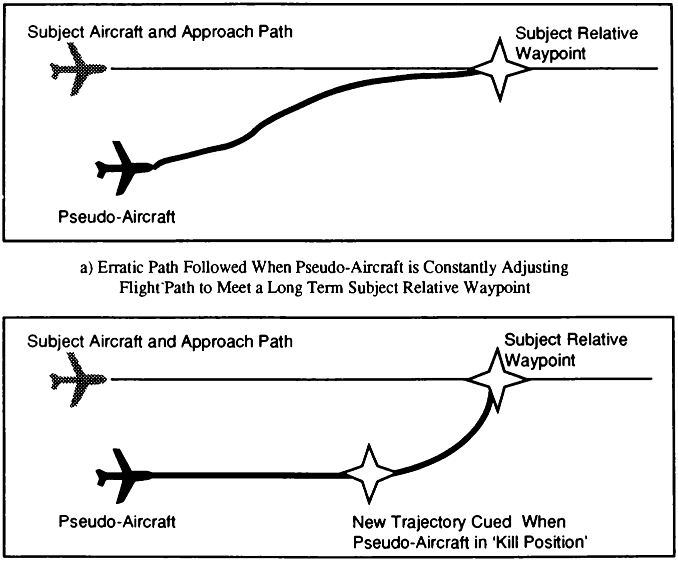
The best performance has been attained by carefully combining these two methods. Figure 6 illustrates an example situation, where a pseudo aircraft is supposed to fly a parallel approach path (apparently to land on a parallel runway), then deviate towards the subject and cause a collision. As shown in Figure 6a, a long time-scale subject relative waypoint can cause an erratic flight path of the pseudo-aircraft; a long time-scale being defined as one in which the subject may alter substantially his or her flight path.

Using two sequences of waypoints, with the transition between them being cued by an amendment, can dramatically improve the final performance. For example, in Figure 6b, the pseudo-aircraft is first steered towards a 'Kill Position' from which it can realistically home on the subject; at that point the pseudo-aircraft is given a new trajectory involving a subject-relative waypoint commanding a collision.

This example provides a formula for creating, reliably, a collision situation and illustrates the need to use subject relative waypoints and amendments together to achieve a high accuracy in the situation generation.

Two considerations do remain, however. First, the use of steering towards, and cueing from, a 'Kill Position' requires prior knowledge in the script or in the RSG logic of the dynamics of a collision.

Second, the requirements of high accuracy recommend the use of subject relative waypoints and frequency amendments; these requirements contradict those previously listed for dealing with high subject observability.


Subject Aircraft and Approach Path
Subject Relative Waypoint
Pseudo-Aircraft
Subject Aircraft and Approach Path
Subject Relative Waypoint
Pseudo-Aircraft
New Trajectory Cued When Pseudo-Aircraft in 'Kill Position'

a) Erratic Path Followed When Pseudo-Aircraft is Constantly Adjusting Flight'Path to Meet a Long Term Subject Relative Waypoint

b) Smoother Path Followed When Pseudo-Aircraft is First Straight to 'Kill Position' and Then an Amended Trajectory Gives an Immediate Subject Relative Waypoint

Figure 6. Comparison of Collision Trajectories Generated With Subject-Relative Waypoints, With and Without Use of 'Kill Position'.

Summary and Conclusions

Flight simulator experiments are being developed or have been completed, testing cockpit traffic displays, issues with Free Flight and datalink implementation, and closely-spaced parallel runway operations. These experiments required interactions with Air Traffic Control and other aircraft to happen consistently and in strictly specified ways.

To meet these needs, Robust Situation Generation was implemented. RSG uses continual feedback of the subject state information for comparison with pre-scripted specifications for experiment situations. It then generates pseudo-aircraft trajectories and Air Traffic Control voice communications which will generate the desired experiment situations despite any reasonable variations of the subject's actions from nominal.

RSG also has the capability to control an arbitrary number of aircraft to an arbitrary level of precision, enabling both more complex situations than can easily be otherwise created and more precise interations. If desired, RSG can incorporate more specific constraints on the situations, allowing for their adjustment to better match the objectives and measurements of the experiment.

Incorporating RSG into flight simulator experiments has yielded consistent results with requiring experimenters to act as pseudo-pilots in real-time during the experiment runs. The experiment design process should consider the capabilities and requirements of RSG at all stages, especially as the scripting of the RSG commands to the pseudo-aircraft and of the Air Traffic Control communications can be a non-trivial, iterative process.

Several variables increase the complexity of generating the RSG scripts: a need for robustness to extreme subject deviations from nominal, a need for high accuracy in the situations, and the ability of the subject to view a great deal of information about the pseudo-aircraft.

Generating the set of pseudo-aircraft commands to cause most experiment situations is an under-determined process. Several different types of RSG commands that can be commonly used to create the same situation have been discussed.

Two factors, however, do tend to constrain the RSG commands. These are: the ability of the subject to perceive the trajectory and intentions of the pseudo-aircraft, and a high desired accuracy in the situations despite subject deviations from their nominal trajectory.

Suggestions for dealing with these factors have been given, such as tailoring the type of trajectory-defining waypoints used to steer the pseudo-aircraft.

These suggestions, however, are often contradictory for cases with both high subject-observability and high accuracy. For example, for high accuracy, the use of subject relative waypoints and frequent amendments is suggested; for maintaining fidelity with high subject observability, few amendments or subject relative waypoints are suggested so that the pseudo-aircraft appear to be following a constant, fixed in space trajectory. Such situations require careful scripting and careful cueing of new pseudo-aircraft trajectories at believable times. The addition of corresponding Air Traffic Control communications can help validate pseudo-aircraft actions at these times.

It should be noted this problem of creating, simultaneously, high fidelity and high accuracy are also present when using other methods of generating pseudo-agents for the subject to interact with, and will become more prevalent as more sophisticated and informative cockpit traffic or collision avoidance systems require testing.

Acknowledgments

This work was supported by the National Aeronautics and Space Administration/ Ames Research Center under grant NAG 2-716.

Pilot Evaluation Results of a Half-Dome Type Simulator System

Hiroyasu Kawahara, Kaoru Wakairo, Akira Watanabe and Kohei Funabiki
National Aerospace Laboratory, Tokyo, Japan

Abstract

In March 1995, the National Aerospace Laboratory installed a helicopter flight simulator in order to study the flying qualities, maneuverability and flight safety of helicopters.

The visual system is particularly important in helicopter flight simulators.

The NAL simulator's visual system comprises a 5 meter radius half-dome screen, six CRT projectors and a high-quality image generator (IG). A small helicopter cockpit is located at the center of the dome.

This paper first describes the function and performance of the half-dome visual system and the image generator. Next, the results of an evaluation of the half-dome visual system by pilots is presented together with the results of an evaluation of the infinity display used in NAL's fixed-wing flight simulator.

The half-dome display hardware was found to be satisfactory, but several points of dissatisfaction were found regarding pilots' sensations of height, speed and depth. With the infinity display, both hardware and pilot sensation were found to be satisfactory.

1. Introduction

In 1995, the National Aerospace Laboratory introduced a helicopter flight simulator for research into helicopter flying qualities, piloting techniques and flight safety(Reference 1).

A helicopter simulator's visual system requires a large field-of-view (FOV) compared with a fixed-wing flight simulator(Reference 2, 3).

High image quality is also particularly important in helicopter simulators compared with fixed-wing simulators. Helicopters spend a lot more time maneuvering close to the ground than fixed-wing aircraft and visual cues such as texture flow and texture element size are important in visually estimating parameters such as speed and height when close to the ground.

A visual system giving high image quality and brightness with a large field-of-view was investigated for the simulator. As a result, it was decided to install a 5 meter radius half-dome screen and to use high-brightness CRT projectors. A new image generator was installed to provide a higher quality image than the existing fixed-wing simulator's original IG, which did not support texturing(Reference 4).

This paper describes the function and performance of the installed system and the results of its evaluation by pilots, researchers and simulator engineers.

2. Computer-Generated Imagery

2.1 Image Generator

A Toshiba TESCO PT-2000SJ image generator, originally designed by Lockheed-Martin, was used. The sample visual scene is shown in Figure 1 and specifications of the IG are listed in Table 1.

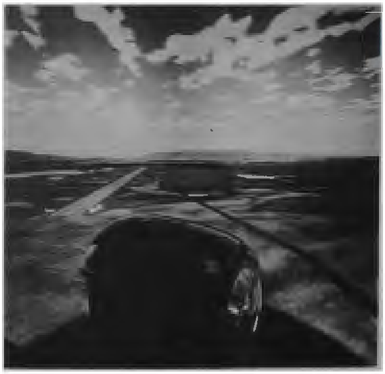


Figure 1. Sample PT-2000SJ Visual Scene

The main features of the image generator installed at NAL are summarized below:

- six channels
- image resolution per channel:
  960 raster scan lines vertically,
  1024 pixels horizontally
- 60Hz update rate
- micro-texture and translucency functions
- up to 128 moving objects

2. 2 Video Signal Distribution

The IG provides three video signal output channels which are distributed to three displays: the half-dome projector system, the fixed-wing simulator's infinity projection system and a multi-channel monitor display (see Figure 2).

3. The Half-Dome Visual Display

A number of factors were examined during the design of the display, including the vertical extent of the field-of-view, the distance from the pilot's eye position to the screen, and the brightness and resolving power of the display. The design of the display is presented below.

3. 1 Required Field-of-View

The pilot's field-of-view from a real helicopter was used to determine that of the simulator. The external view from a Eurocopter AS-350 helicopter had been measured in a previous study into pilot visual cues (Reference 5) and is shown in Figure 3. The view from the pilot's eyepoint is about 50 degrees upward, 60 degrees downward and in excess of 180 degrees horizontally between left and right. This was used as a basis for the design of the half-dome display.

3. 2 The Half-Dome Screen

3. 2. 1 Radius and Field-of-View

Most dome-type simulators in Japan use a full-dome screen of approximately 3.5-4.5 meters radius. However, a study has shown that this type of dome simulator can easily induce motion sickness (Reference 6), one of the reasons being that the screen is too close to the eye position of the pilot.
In our own experience of several years ago, a pilot experienced simulator sickness when his eye position was close to the screen. The sickness was particularly pronounced when the distance was less than 2 meters, but somewhat relieved when the distance was greater than 5 meters. On the basis of this evidence, it was felt that a large dome radius might be effective in reducing simulator sickness, but of course this means that the dome occupies more space.

Item	Specification
Display channels	6 ch
Resolution	1024(H) × 960(V)
Raster	Interlace
Data update rate	60 Hz
Frame refresh rate	30 Hz
Video channels	3 ch
Textuer Maps	96 full color(256×256) Microtexture
Transport Delay	67 ms
View points	6
Moving objects	28
Special Effects	Translucency Weather Effect Height above Terrain Collision Detection Others

Table 1. Technical Specifications of PT-2000SJ Image Generator

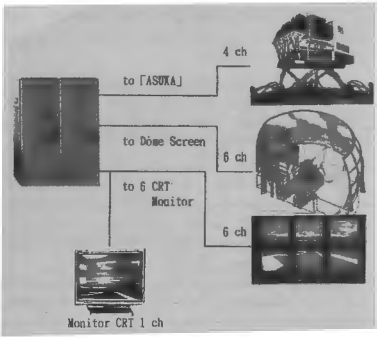


Figure 2. Schematic of Video Signal Distribution

The size of the room in which the simulator was to be installed constrained the maximum radius of the half-dome screen to 5. 5 meters, so the half-dome screen was designed to have a 5 metre radius to give a 0. 5 meter margin. The height of the room constrained the maximum vertical extent of the field-of-view to 80 degrees. In helicopter take-off and landing, a large lower FOV is particularly important, so the vertical FOV was designed to be 30 degrees upper and 50 degrees lower. The horizontal FOV was designed to be 180 degrees. Figure 4 shows an external view of the half-dome.

3. 2. 2 Screen Gain

Screen paint can be chosen to give a screen gain of between 0. 5-4. 0. Table 2 shows the relationships between screen gain and display characteristics. Paint gain affects the brightness, contrast and directionality of the screen:

- paint gain is directly related to screen brightness; a low paint gain gives low screen brightness and vice versa.

- lower paint gains give more contrast, and so more "depth" to the scene; a high paint gain gives a monotone picture.

- with lower paint gains, the directional tendency is broader and the picture can be seen clearly from a wider range of positions.

Initially, a screen gain of 2 was assumed. A computer was then used to simulate the variation in brightness across the screen surface with screen gains of 1. 8 and 1. 5. The results of the simulation, shown in Figure 5, indicated that a screen gain of 1. 5 would give less variation in brightness across the display surface.

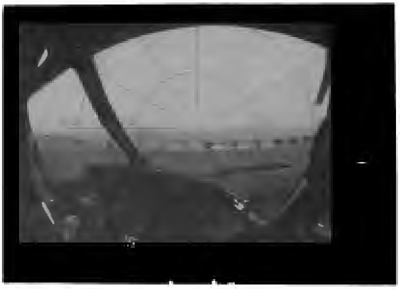


Figure 3. External View Angle from an AS-350 Helicopter

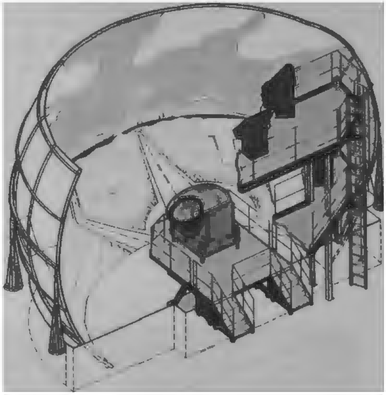
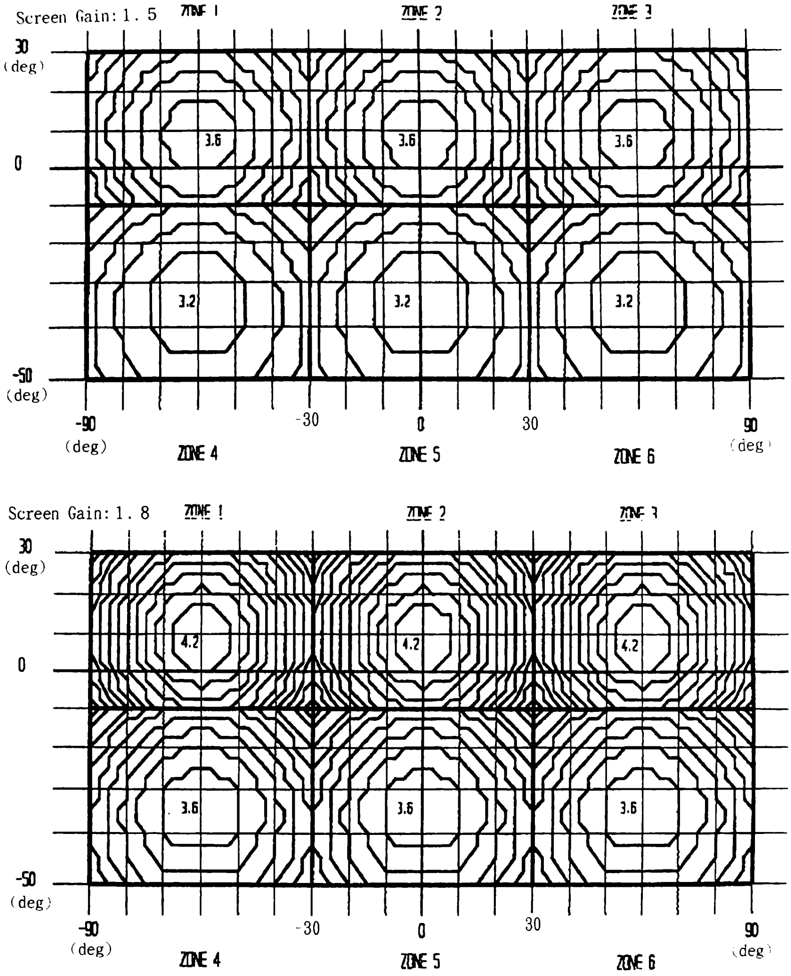


Figure 4. External View of Half-Dome Screen

Item	Low Screen Gain	High Screen Gain
Directionality	Broad:Viewing angle is wide	Narrow:Viewed close to screen axis
Brightness of Screen	Low brightness(Dark)	High Brightness(light)
Continuity for multiple projectors	Good	Poor
Quality of Visual Scene	High Contrast and High Quality	Low contrast and monotone
Completion of screen	Rough (because directionality is broad)	Fine (because directionality is narrow)

Table 2. Relationships between Screen Gain and Visual Scene



ALL CONTOUR BANDS 0. 2 ft-LAMBERT INCREMENTS

Figure 5. Simulation Result of Brightness Maps

Since a high brightness projector could be obtained, it was decided to use a screen gain of 1. 5 to give less variation in picture brightness, trading this against overall brightness.

3. 3 Visual Projection System

3. 3. 1 Scene Brightness

The projection system was required to give a white peak value at the screen surface of over 3. 5 ft-Lamberts. A high brightness projector was necessitated by the selection of the lower screen gain of 1. 5 and the drop-off in brightness due to the comparatively large projection distance of 5 meters. It was decided to use a high brightness 9-inch three tube CRT projector to meet this criterion, and a projector made by the British company SEOS was selected. This projector's nominal performance guaranteed a peak of 3. 5fL at the centre of the screen.

With the projector installed, the white peak values were measured, and the result is shown in Figure 6. At the centre of the screen, the maximum peak value is 3. 36fL, and at the edge the lowest value is 0. 8fL. This non-uniformity of brightness was one of the parameters evaluated by pilots.

3. 3. 2 Image Resolution

The image generator resolution is 1024 pixels horizontally by 960 scanlines vertically. The maximum discrimination power of the human eye for a black and white stripe is about 1 minute of arc (0. 017 degrees)(Reference 7).

The relationship between screen size and vertical discrimination of the projected image was investigated. If two projector stages are used vertically, the vertical resolution is 0. 041 degrees (that is, one scanline subtends 0. 041 degrees at the display surface). If three stages are used, the resolution is 0. 029 degrees.

Although a three-stage system yields a resolution closer to the human eye's discrimination limit, a two-stage system was adopted due to room size constraints and cost-performance tradeoffs. The resolving power is thus about two times worse than the optimum case.

For horizontal coverage, three projector stages were found to be sufficient. The half-dome display therefore uses six projectors in all.

3. 3. 3 Image Distortion Correction and Edge Blending

The projection of a flat image onto a curved surface introduces distortions which must be corrected. Further, since six projectors are used to display the image, it is necessary to blend the edges of their coverage areas together smoothly.

There are two methods for correcting distortion; either performing the correction in the image generator or by correcting the source image at the projection stage. The disadvantage of the former case is that the IG must generate a source image containing corrective distortions. The image distortion was therefore corrected at the projection stage, which also gave an easy solution to the edge blending problem.

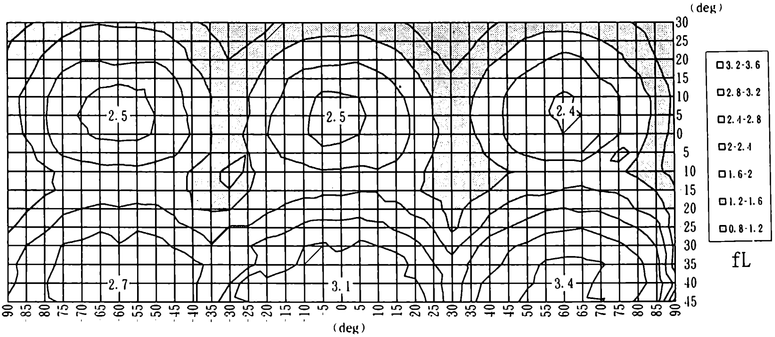


Figure 6. Brightness Masurement Results of Harf-Dome Screen

The SEOS DDHU projection system has the necessary capability for correlation and edge blending via a remote control, and the corrections can be made very easily. This was another factor in the selection of this particular projection system.

4. Evaluation of the Visual Display System

Two evaluations were conducted concurrently to test whether the functionality and performance of the half-dome display were suitable for a helicopter simulator visual system, and whether the infinity display was suitable for the fixed-wing simulator visual system.

The evaluations were of subjects' opinions regarding the physical characteristics of the display as well as the sensations induced by the display.

4. 1 Subjects

The evaluation was carried out primarily by pilots and engineers working for Japanese aircraft manufacturers, with NAL pilots and researchers also participating (see Table 3).

4. 2 Evaluation Method

Each subject was required to evaluate both the infinity display and the half-dome display. In the case of the half-dome display, a Super Puma helicopter flight dynamic model was used. The fixed-wing simulator simulates the ASUKA STOL research transport aircraft.

For each evaluation test, a questionnaire sheet (Table 4) was presented to each subject prior to the test, and filled in during the evaluation. Subjects' comments were recorded on the same sheet following the test.

Subjects were required to evaluate the following:

- hardware
- FOV
- display brightness
- image blending
- colour balance among channels
- brightness balance among channels
- sensations of:
    - height
    - depth
    - speed

Subjects	Pilots/Engineers or Researchers
Aircraft Maker(JPN)	
K Company	2 ／ 3
M Company	5 ／ 2
F Company	4 ／ 0
S Company	1 ／ 0
NAL	2 ／ 4
Other Organization	2 ／ 0

Table 3. Subject Population

For each parameter, subjects were required to indicate their evaluation by placing a mark on a continuous scale between two extremes. The position of the mark along the scale was then converted into a score from 0 to 100, where "bad" or "insufficient" is 0, "ordinary" is 50 and "good" or "sufficient" is 100.

4. 3 Results

Figure 7 shows the average scores recorded for each parameter for the two evaluation cases (half-dome and infinity displays). The centre of the circle is 0 and the edge is 100.

4. 3. 1 Half-Dome Display

As can be seen from Figure 7a, the opinions of the pilots regarding the hardware were good or very good (80% or better), but the quality of some of the induced sensations were not as good as had been hoped.

4. 3. 2 Infinity Display

The fixed-wing simulator's original infinity display visual system was installed with the simulator itself 15 years ago, but it was decided to upgrade the visual system and to evaluate and compare it with the half-dome display.

As can be seen from Figure 7b, the results regarding display hardware were good or very good, and good or fairly good regarding induced sensations. The results are similar to those obtained with the original visual system, which lacked features such as texture mapping.

New Visual System Evaluation Sheet

Date :__________, 1995

Name :________________ Name of Organization or Company :________________

Experience of Aircraft and Flight Simulator :
Fixed Wings________Y (________Hr) , Simulators________Y (________Hr)
Helicopters________Y (________Hr) , Simulators________Y (________Hr)

1 . Harf- Dome Type Visual Display System for Helicopter Simulator

①Field of View	Up-side	Sufficient	ordinary	insufficient
	Down-side	Sufficient	ordinary	insufficient
	Horizontal	Sufficient	ordinary	insufficient
②Brightness		Sufficient	ordinary	insufficient
③Scene Blending		Sufficient	ordinary	insufficient
④Color Balance among channels		Sufficient	ordinary	insufficient
⑤Brightness Balance among channels		Sufficient	ordinary	insufficient
⑥Height Sensation		Sufficient	ordinary	insufficient
⑦Depth Sensation		Sufficient	ordinary	insufficient
⑧Speed Sensation		Sufficient	ordinary	insufficient

2 . Infinity Display for Fixed-Wing Simulator

①Brightness	Sufficient	ordinary	insufficient
②Color Balance	Sufficient	ordinary	insufficient
③Continuas Visual Scene	Sufficient	ordinary	insufficient
④Color Balance among channels	Sufficient	ordinary	insufficient
⑤Brightness Balance among channels	Sufficient	ordinary	insufficient
⑥Height Sensation	Sufficient	ordinary	insufficient
⑦Depth Sensation	Sufficient	ordinary	insufficient
⑧Speed Sensation	Sufficient	ordinary	insufficient

Table 4. Example of Visual System Evaluation Sheet

4. 3. 3 Subject Comments

Subject comments are summarized as follows:
1) There are some discrepancies between instrument-indicated height and speed values and the sensations of these afforded by the visual system.
2) Information necessary for judgement of height and distance from geographical references is poor.
3) Subjects were generally unaware of projector blending when shown a moving image, but were aware of the blending if a static image was displayed.
4) There were no instances of simulator sickness.

4. 4 Discussion

4. 4. 1 Half-Dome Display

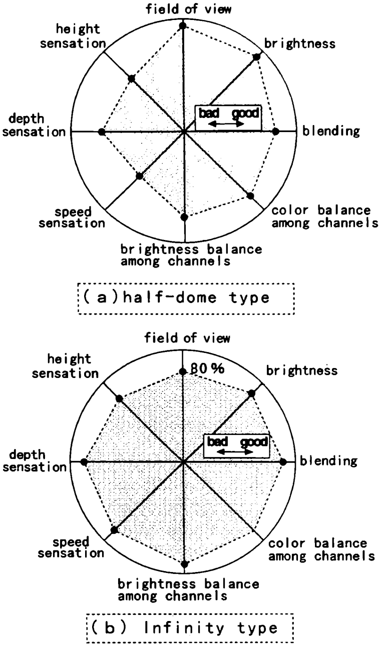
In helicopter simulation, the most important flying qualities to evaluate are those when flying close to the ground. In this environment, the pilot must be able to see the ground surface clearly.
Image texturing was used to increase realism, but due to restrictions in the image generator, this cannot be applied to all ground locations. During the simulation, pilots made negative comments when flying over surfaces lacking fine texture detail.
Further, hills and mountains are represented using phototexturing, which are very realistic when viewed from a distance or high altitude, but as they are approached, they become fuzzy and pilots lose the sensation of height.

In order to improve realism, it is recommended that:
1) Micro-textures are used to represent the ground effectively
2) The number of buildings and moving objects (e. g. other aircraft, road vehicles etc.) which can be seen by the pilots is increased.

There were no reports of simulator sickness. It is speculated that this may be due to the large radius of the dome (hence large distance between the screen and the pilot's eyepoint) and the high brightness and resolution.

4. 4. 2 Infinity Display

The infinity display was evaluated using a fixed-wing transport aircraft simulator, where little time is spent flying close to the ground except in the cases of take-off and landing.



(a) half-dome type

(b) Infinity type

Figure 7. Evaluation Results

The visual database on and around the runway is highly detailed; runway landing-zone and center line markings can clearly be seen. For these reasons, there were no unfavourable comments regarding the infinity display, and no problems reported with sensations of speed or scene depth.

5. Applications of the Half-Dome Display

The half-dome display has a very large FOV, and so has been used for other applications. Some examples are presented below.

5.1 Fire Fighting by Helicopter

The half-dome simulator was used in a simulation of fighting a fire in a high-rise building by helicopter. The aim was to discover and elucidate technical problems such as wind phenomena around buildings and to establish techniques for fighting fires in building using helicopters(Reference 8).

For the simulation, the cockpit center line was fixed at 30 degrees left of the screen centre to enlarge the right field-of-view. The wide FOV afforded by the half-dome display was therefore particularly useful in this instance as well as being useful for general helicopter simulation. Figure 8 shows a scene from a simulation. Here, the pilot can see how the fire is diminished by the position of the simulated water jet.

Pilots commented that it was very easy for them to control the helicopter using the wide-angle display, but they found it difficult to judge distance from the building and height in the hover using purely simulated visual cues. However, these factors were not considered significant for the purposes of this research.

5.2 Space Sickness Simulation

To examine the sense of equilibrium in medical science or psychology, a small bowl-type screen or a cylindrical rotating screen is typically used as a visual display. However, the size of the visual scene displayed by these devices is small and limited. The large FOV afforded by the half-dome display enables experiments to be conducted with much higher accuracy.

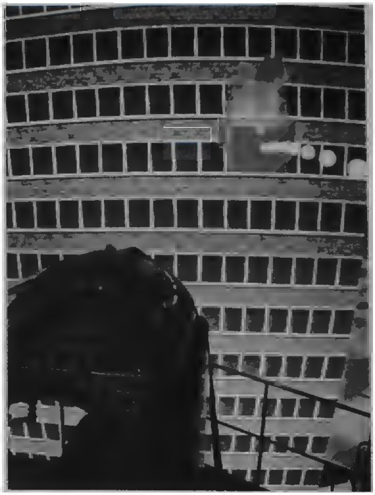


Figure 8. Visual Scene Example of a Fire in a High-Rise Building

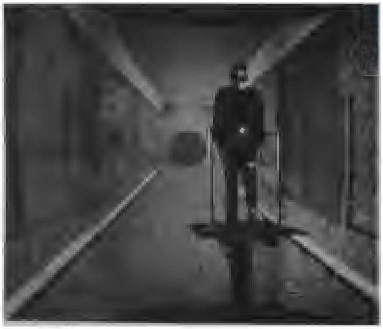


Figure 9. Visual Scene Example of Space Sickness Experiments

NAL conducted an experiment in cooperation with NASDA investigating vertigo and other illusions in a space module. The interior of the Japanese Experiment Module (JEM) was displayed on the half-dome screen (Figure 9). The real JEM module is 5 metres wide, 10 metres long and 2 metres high. During the experiment, the experience of length of the module was felt to be realistic, but the experimenter felt the width to be larger than that of the real JEM module. This has shown that in the case of a simulation where the real distance from the eye position to the object is less than 5 metres, the half-dome display is not always useful.

5. 3 Trembling Body Measurement

Several visual display devices have been used in studies of rehabilitation of the physically handicapped and brain functional disorders, including Head-Mounted Displays.

Thus far, almost all devices used in such research have had only narrow fields of view and poor resolution. Researchers are therefore looking for a suitable device with a wide field of view.

This year, an experiment was conducted in this field using the half-dome display, which has a wide FOV and high resolution. The experiment found that peripheral vision affects the sensation of equilibrium in a subject, and a trembling sensation may be induced.

6. Conclusion

This paper first described the function and performance of a half-dome display and its associated image generator. Next, the results of an evaluation by pilots were reported.

The system has been employed to good effect in helicopter simulation and related research areas. In future, there are plans to modify the image generator's database and to use the display in other applications.

This year a small 6 degrres-of-freedom motion platform will be installed in the half-dome, and its effectiveness will be investigated.

Finally, we would like to thank the engineers from Toshiba TESCO who installed the new display and we would also like to thank the pilots and engineers who participated in the evaluation tests.

References

1) Wakairo, K. et al. , Functions of NAL Fixed Base Simulator for Helicopter Research. Aircraft Symposium of JSASS, 1992.

2) Kawahara, H. et al. : The Functions and Characteristics of NAL Flight Simulator Cockpit Systems , NAL TM-577. 1987. (In Japanese.)

3) Watanabe, A. :Advanced System Technologies of NAL Flight Simulator. NAL Reseach Progress. 1995.

4) Wakairo, K. et al. : The Functions and Characteristics of NAL Flight Simulator Visual Systems, NAL TM-581, 1988. (In Japanese.)

5) Kawahara, H. et al. : A Field Study on Visual Cues of Helicopter Pilots, NAL TM-684, 1995. (In Japanese.)

6) Muraoka, H. et al. : The effect visual cue in Simulator Sickness, Proceedings of the 32nd Aircraft Symposium, 1994. (In Japanese.)

7) Wada, Y. et al. : A Handbook of Sensory and Perception, Seishin Syobo, 1969. pp. 251 -252. (In Japanese.)

8) Funabiki, K. et al. : The Operational Simulation Experiments of a Fire Fighting Helicopter, 27th JSASS Annual Meeting, 1996. 4. (In Japanese.)

AIAA-96-3556-CP

PASSIVE NAVIGATION FROM IMAGE SEQUENCES: A PRACTITIONER'S APPROACH

Adam X. Miao[*], Greg L. Zacharias[†]
Charles River Analytics
Cambridge, MA 02138
and
Rik Warren[‡]
Armstrong Laboratory
Wright-Patterson Air Force Base, OH 45433

ABSTRACT

This paper presents a systematic solution to passive navigation focusing on the efficiency, robustness, and accuracy of the solution. A three-stage motion and terrain shape estimator is developed, consisting of: 1) an image processor; 2) a snapshot motion and terrain shape estimator; and 3) a Kalman filter to smooth the snapshot estimate. The first stage image processor computes image measurements (spatiotemporal gradients) each time a new image is acquired. Using only the current image measurements, the snapshot estimator then generates a "snapshot" (instantaneous) motion and terrain shape estimate of limited accuracy. A Kalman filter is employed in the last stage to improve the accuracy of the snapshot estimates by integrating information over time and exploiting the continuity constraints in the observer's motion and in the terrain. Driving the estimator with computer generated imagery and flight-recorded imagery, we generate running estimates of altitude and attitude with respect to the overflown terrain. The results indicate that the estimator is capable of accurate, efficient, and robust motion and terrain shape estimation.

I. INTRODUCTION

Passive navigation and many other computer vision applications require recovering both the observer motion and the terrain shape from a sequence of dynamic images of the stationary terrain. Many solutions exist to this problem. Unfortunately, from a practitioner's point of view, the problem is still mostly unsolved since these solutions either: 1) adopt computationally intensive and numerically unstable solution techniques; 2) assume the terrain structure or motion as known; or 3) require the specification of problem-dependent parameters that can only be determined via trial and error.

A common thread of these solutions is to formulate the passive navigation problem as a two-stage process: first, compute the optical *flow-field* based on the dynamically changing images; second, estimate the motion states and the relative depth/shape based on the computed flow and possibly on a model of the viewed terrain. This two-stage approach provides a convenient *divide and conquer* mechanism. One group of researchers can thus concentrate on flow-field computation without considering its consequences for motion estimation, while another group can concentrate on the estimation of motion states and terrain shape, assuming availability of an accurately computed flow-field without worrying about the feasibility of its generation.

There are a number of algorithms for computing flow-fields, given a sequence of dynamically changing images. These algorithms can be roughly grouped into feature-based, gradient-based, and frequency-based methods. Table 1 summarizes the advantages and disadvantages of each method.

Table 1: Advantages and Disadvantages of Various Flow-Field Computation Methods

Method	Advantages	Disadvantages
Feature-based	Allow large inter-frame time Lowest computation load	Difficulty in determining & matching features Sparse flow-field Not suitable for featureless imagery
Gradient-based	Easy implementation Wide applicability	Require small inter-frame time and smooth image Worst performance High computation load
Frequency-based	Best performance	Highest computation load A priori imagery information required Large number of frames

Feature-based methods extract flow-fields by matching features from one image frame to the next and determining their shifts [1,20,22]. The method has the advantage of the lowest computational requirement among the three approaches, since flow-fields are only computed for the sparse set of features in the image and shift computation involves only a few arithmetical operations. Moreover, it allows arbitrary but known inter-frame time between images. The method, however, has the disadvantage of great difficulty in determining and matching invariant features from one image to the next due to uncertainties caused by occlusion, observer motion, and image sensor noise. Moreover, it provides only a sparse subset of the entire flow-field that depends on the number of recognizable features in an image, and cannot be used with a featureless scene.

Gradient-based methods compute flow-fields using spatiotemporal gradients of (filtered or original) image intensity at each pixel [10,14,17]. The method has the advantage of ease of implementation in both sequential and parallel computers since computation of spatiotemporal gradients at a pixel is simple and involves only a few neighbor pixels. The method also has the

[*] Senior Scientist

[†] Principal Scientist, Senior Member AIAA

[‡] Engineering Research Psychologist

advantage of wide applicability to both feature-rich and featureless images. Three major disadvantages, however, prevent the method's usage. First, the method is known to have the worst performance in the case of non-smooth images due to occlusion boundaries and perceptual distortion. Second, since the gradient-based method usually must use an iterative computation method to improve the quality of the computed flow-field, the computational load for the approach can be high. Third, the flow-field computation needs the specification of the scene-dependent parameters.

Frequency-based methods produce flow-fields using velocity-tuned filters in the Fourier domain 5,6,7. By using large numbers of images and rather stable frequency domain image features (energy or phase), the method has the advantage of generating the most accurate flow-field. The method, however, has the highest computational load, requires *a priori* motion and image information in designing the filters, and is the most difficult to implement since the filter parameters must be tuned according to the image spectrum.

Feature-based and frequency-based methods are not suitable for passive navigation applications, which require real-time operation, need the methods to work for both feature-rich and featureless scenes, and usually have no a *priori* information on motion and terrain. These conditions make gradient-based methods a remaining candidate for passive navigation. Unfortunately, gradient-based methods fail to produce a computed flow-field that has a relative error of less than 10% — an error threshold required for reliable motion and terrain estimation according to Barron, Jepson & Tsotsos [3]. Barron, Fleet & Beauchemin [2] conducted quantitative evaluation of the three methods. Their results show that only well-tuned frequency-based algorithms could achieve the required accuracy, whereas both feature-based and gradient-based algorithms had average errors close to or above 10% even for synthetic (computer-generated) flat-terrain images. The same above-10% average errors were also reported in quantitative evaluations of feature-based and gradient-based algorithms on two relatively simple synthetic image cases in Little & Verri [12].

After flow-fields are computed (presumably or actually), a number of algorithms exist to estimate motion/shape parameters. These algorithms can be roughly grouped into instantaneous (snapshot) and Kalman filter estimators. Of the instantaneous estimators, Zacharias, Caglayan & Sinacori [24] developed a quasi-static estimator to estimate an observer's instantaneous heading (vertical and lateral), angular velocity, and relative depth or impact *times* (the speed-scaled range or 3D shape) by decomposing the motion state and terrain shape estimation problems. Heeger and Jepson [8] split the flow-field equation via algebraic manipulation into three sets of equations. The first set relates the flow field to only the translational (heading) motion. Thus, relative depth and rotation motion (angular velocity) need not be known prior to estimating the translational motion. Once the translational motion has been determined, the second set of equations relating to both translational and rotational motion is then used for estimating rotation. Finally, the depth at each pixel is estimated using the third set of equation, given the estimated translation and rotation.

Kalman filter based methods use the computed flow-field vectors as observations, and integrate them with the motion/shape parameters over multiple image frames to improve motion/shape estimation accuracy. Various schemes have been developed.[4,15,22,23]. Two fundamental problems here are: 1) the need to track and establish correspondence over a larger numbers of image points or features that appear/disappear or occlude each other; and 2) the numerical instability of the filtering process and the high computational cost due to the high dimensional nonlinear dynamics of the motion/shape parameters. These have restricted the application of Kalman filtering mostly to the use of feature-based flow-fields. Even in these cases, accurate motion states must be known to achieve promising estimation results in dealing with real images [22]. Extension of the Kalman filter approach to the nonfeature-based flow-fields was proposed by Matthies et al. [15], but accurate motion states were again assumed to be known.

Striving to overcome difficulties in accurately and efficiently estimating flow-fields by the gradient-based method but keeping its other advantages, Horn & Weldon [11], Heel & Negahdaripour [9], Negahdaripour & Lee [18], and Zacharias, Miao, & Warren [26] have begun to develop various direct methods that estimate motion and terrain shape using image gradients directly. These methods either require that the motion is purely translational/rotational [11], the motion is constant over time [9], the terrain contains special structures [18], or the terrain is constant over time [26]. Moreover, most of these methods either provide only instantaneous estimates or employ complicated extended Kalman filters.

This paper presents a systematic and practical solution to passive navigation, focusing on the simplicity, efficiency, robustness, and accuracy of the solution. A three-stage motion and terrain shape estimator is developed, consisting of: 1) an image processor; 2) a snapshot motion and terrain shape estimator; and 3) a Kalman filter to smooth the snapshot estimate. The first stage image processor computes image measurements (spatiotemporal gradients) each time a new image is acquired. Using solely the current image measurements, the snapshot estimator then generates a "snapshot" (instantaneous) motion and terrain shape estimate of limited accuracy. A Kalman filter is employed in the last stage to improve the accuracy of the snapshot estimates by integrating information over time and exploiting the continuity constraints in the observer's motion and in the terrain. Driving the motion and terrain shape estimator with computer generated imagery (CGI) and flight-recorded imagery, we generated running estimates of altitude and attitude with respect to the overflown terrain. The evaluation results indicated that the estimator is capable of accurate, efficient, and robust motion and terrain shape estimation.

The remainder of this paper is organized as follows. Section II describes the basis for passive navigation: an image gradient-based estimation constraint equation. Section III gives a technical description of each component of the three-stage motion and terrain shape estimator. Section IV evaluates overall operation of the estimator for both Computer Generated Images (CGI) and real terrain imagery. Finally, section V provides a summary of the paper.

II. ESTIMATION CONSTRAINT EQUATION

This section derives an image gradient-based estimation constraint equation. The objective is to explicitly relate unknown motion/terrain variables to directly measurable (computable) image gradients, using three key image, motion, and terrain relations.

Consider an observer undergoing both translational and rotational motion over motionless terrain as illustrated in figure 1. The observer's linear and angular velocity are denoted by the vectors **V** and **Ω**, respectively. Let P_i be a viewed point on the terrain, located at a range r_i from the observer. In the observer's frame of reference, the unit-length line-of-sight (LOS) direction vector $\mathbf{u}_i = \rho_i/|\rho_i|$ will appear to change with time, at a rate given by

$$\dot{\mathbf{u}}_i = \boldsymbol{\omega}_i \times \mathbf{u}_i \tag{1a}$$

where $\boldsymbol{\omega}_i$ is the rotation rate of the LOS vector (effectively, the 3D visual flow seen by the observer), and is shown by Zacharias and Levison [25], to be given as

$$\boldsymbol{\omega}_i = \mathbf{u}_i \times (\mathbf{u}_i \times \boldsymbol{\Omega}) - \frac{1}{\tau_i} (\mathbf{u}_i \times \mathbf{u}_V) \tag{1b}$$

where $\mathbf{u}_V$ is the observer's *heading vector*, and τ_i is the *impact time*, defined by

$$\mathbf{u}_V = \frac{\mathbf{V}}{|\mathbf{V}|}, \qquad \tau_i = \frac{\rho_i}{|\mathbf{V}|} \tag{2a,b}$$

The unit length heading vector defines only the direction of the observer's motion, but not his speed. The impact time is the elapsed time before the observer impacts with the surface at point P_i, *if* the observer were to head directly at P_i, at the speed $|V|$.

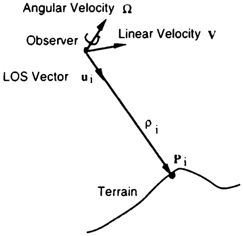


Fig. 1: Line-of-Sight Rate

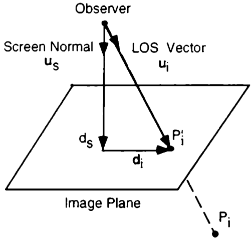


Fig. 2. Image Plane Geometry

The first key relationship relates the 3D visual flow to its 2D projection on a hypothesized image plane. Consider the 2D image of the terrain formed on an image plane within the observer's field-of-view (FOV), as illustrated in figure 2. Let the focal point be the origin of a screen coordinate system. Let $\mathbf{u}_S$ denote the unit length vector normal to the sensor plane, and d_S denote the effective focal length. The image of point P_i, given by P_i', located by the vector $\mathbf{d}_i$ in the image plane, is defined by the focal-length normalized x-y coordinates $\boldsymbol{\delta}_i = \mathbf{d}_i/d_S = (x_i, y_i, 1)^T$. As shown by Zacharias, Miao, and Warren,[26] the rotation rate of the LOS vector $\mathbf{u}_i$ generates the following rate of change of this in-plane location vector $\dot{\boldsymbol{\delta}}_i$:

$$\dot{\boldsymbol{\delta}}_i = \frac{\boldsymbol{\omega}_i \times \mathbf{u}_S}{(\mathbf{u}_i \bullet \mathbf{u}_S)^2} = \mathbf{H}_{i\Omega}\boldsymbol{\Omega} + \mathbf{H}_{iv}\frac{\mathbf{u}_v}{\tau_{iz}} \tag{3a}$$

where $\tau_{iz} = \tau_i / |\boldsymbol{\delta}_i| = \tau_i / \sqrt{1 + x_i^2 + y_i^2}$ is the plane normal (screen coordinate system) impact time, $\dot{\boldsymbol{\delta}}_i$ is a two dimensional vector in the screen x-y plane, and

$$\mathbf{H}_{i\Omega} = \begin{pmatrix} x_i y_i & -(x_i^2 + 1) & y_i \\ (y_i^2 + 1) & -x_i y_i & -x_i \end{pmatrix} \tag{3b}$$

$$\mathbf{H}_{iv} = \begin{pmatrix} -1 & 0 & x_i \\ 0 & -1 & y_i \end{pmatrix} \tag{3c}$$

The second key relationship relates 3D visual flow to the instantaneous spatial and temporal image gradients in the 3D image plane. Using the partial derivative rule to compute the intensity change of image point P_i with time at an in-plane location $\boldsymbol{\delta}_i$, we obtain the following expression for the total temporal derivative of image intensity:

$$\frac{dI(x_i, y_i, t)}{dt} = I_{ix}\frac{dx_i}{dt} + I_{iy}\frac{dy_i}{dt} + I_{it} \tag{4}$$

where $I_{ix} = \partial I_i/\partial x$ and $I_{iy} = \partial I_i/\partial y$, and we have assumed the image brightness function is first order continuous at the image point $\boldsymbol{\delta}_i$. Assuming that the brightness of any terrain point is constant over time, we require the total derivative of (4) to be zero, so that

$$I_{ix}\frac{dx_i}{dt} + I_{iy}\frac{dy_i}{dt} + I_{it} = 0 \qquad \text{or}$$

$$I_{it} = -(I_{ix}, I_{iy})\dot{\boldsymbol{\delta}}_i \tag{5}$$

where $\dot{\boldsymbol{\delta}}_i$ is given by (3a). Substituting (3a) into (5), for each pixel i we obtain an *estimation constraint equation* that relates the directly measurable image intensity I_i (through its gradients I_{it}, I_{ix}, I_{iy}), with the observer motion states $\boldsymbol{\Omega}$, $\mathbf{u}_V$, and terrain impact time τ_{iz} as follows:

$$I_{it} = \mathbf{h}_{i\Omega}^T\boldsymbol{\Omega} + \frac{1}{\tau_{iz}}\mathbf{h}_{iv}^T\mathbf{u}_v \tag{6a}$$

where

$$\mathbf{h}_{i\Omega} = \mathbf{H}_{i\Omega}\begin{pmatrix} I_{ix} \\ I_{iy} \end{pmatrix} \qquad \text{and} \qquad \mathbf{h}_{iv} = \mathbf{H}_{iv}\begin{pmatrix} I_{ix} \\ I_{iy} \end{pmatrix} \tag{6b}$$

III. THREE-STAGE MOTION AND TERRAIN SHAPE ESTIMATOR

Figure 3 shows the three-stage motion and terrain shape estimator that is composed of an image processor, a snapshot estimator, and a Kalman filter. The image processor computes image measurements each time a new image (frame) is acquired at the image sensor. We simulated real-time imaging operations using CGI and pre-recorded imagery. The image measurements consist of first order image gradients and results from arithmetic operations between the image gradients and the image pixel coordinates. Based solely on the current image measurements, the snapshot estimator then generates a "snapshot" estimate of motion states and terrain shape. Using the snapshot estimates as state measurements, the Kalman filter improves estimation accuracy by integrating information over time and exploiting the continuity constraints in the observer's motion states and the terrain shape. In the following, subsection A describes the image processor, subsection B the snapshot estimator, and section C the Kalman filter.

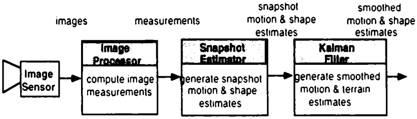

images
measurements
snapshot motion & shape estimates
smoothed motion & shape estimates
Image Sensor
Image Processor
compute image measurements
Snapshot Estimator
generate snapshot motion & shape estimates
Kalman Filter
generate smoothed motion & terrain estimates

Fig. 3. Three Sage Motion and Terrain Shape Estimator

A. Image Processor

The image processor computes the image measurements $\{\mathbf{h}_{iW}, \mathbf{h}_{iv}, I_{it}\}$ each time a new image is acquired. The processor boils down to computing the image gradients, since computing $\{\mathbf{h}_{iW}, \mathbf{h}_{iv}\}$ via (3b) involves only arithmetic operations between image gradients and image pixel coordinates. We use a direct numerical differentiation method to compute the image gradients. Specifically, Let k be the current time (image frame) index, and Δt be the frame interval. At the current time k, we have image k defined by its intensity map $\{I(x_i, y_i, t)\}$, where index i denotes the ith pixel in the image. The image gradients are computed for each pixel using the following two-point central difference formulas:

$$I_{ix} = \frac{I(x_i+\Delta x,\, y_i,\, t-\Delta t) - I(x_i-\Delta x,\, y_i,\, t-\Delta t)}{2\Delta x}$$

$$I_{iy} = \frac{I(x_i,\, y_i+\Delta y,\, t-\Delta t) - I(x_i,\, y_i-\Delta y,\, t-\Delta t)}{2\Delta y}$$

$$I_{it} = \frac{I(x_i,\, y_i,\, t) - I(x_i,\, y_i,\, t-2\Delta t)}{2\Delta t} \tag{7}$$

where Δt is the temporal sampling time, and Δx and Δy are the x and y pixel dimensions. The two-point central formula provides higher accuracy than the standard two-point backward formula ($O(\Delta x)^2$ versus $O(\Delta x)$), at the cost of storing one additional old image frame. The memory cost of this is relatively low. Moreover, since the temporal and spatial gradients are estimated at the same (delayed) point in time, no phase shift is introduced. After I_{ix} and I_{iy} are computed, the image measurements $\mathbf{h}_{iW}$ and $\mathbf{h}_{iv}$ are computed using (3b), (3c) and (6b).

Computation of image measurements using this algorithm requires 13N floating point operations (flops), where 3N flops are for gradient computation, and 10N flops are for image measurement computation. A flop roughly constitutes the effort of doing a floating point add, a floating point multiply, and some indexing. Image measurement computation can be done in parallel since computation for each pixel can be performed independently. Consequently, if a P processor parallel computer is available for the computation, a computation reduction of P times can then be achieved.

B. Snapshot Estimator

The snapshot estimator determines the snapshot (instantaneous) motion and terrain estimates $\hat{\mathbf{\Omega}}$, $\hat{\mathbf{u}}_v$,and $\{\hat{\tau}_{iz}\}$ that minimize the least square residue of the estimation constraint equation [26]

$$r(\mathbf{\Omega}, \mathbf{u}_v, 1/\tau_i) = \sum_{i=1}^{N} \left| I_{it} - \mathbf{h}_{i\Omega}^T\mathbf{\Omega} - \frac{1}{\tau_i}\mathbf{h}_{iv}^T\mathbf{u}_v \right|^2 \tag{8}$$

using N measurements $\{\mathbf{h}_{iW}, \mathbf{h}_{iv}, I_{it}\}$ generated from the image processor. Direct inspection of (8), however, reveals that it is under-constrained since there are N constraints for N + 5 variables (N unknown impact times and five unknown motion states).

We resolve this "under-constrained" difficulty by requiring that the terrain impact times (relative terrain geometry) satisfy a polynomial function. Specifically, we model the inverse of the z-axis terrain impact time τ_{iz} as an n-th order polynomial of the in-plane image coordinates (x_i, y_i), given by

$$\frac{1}{\tau_{iz}} = \mathbf{c}^T\mathbf{f}_i \tag{9a}$$

where

$$\mathbf{c} = (c_0, c_2, \cdots, c_{n^2+3n/2})^T \tag{9b}$$

$$\mathbf{f}_i = (1, x_i, y_i, \cdots, x_i^n, x_i^{n-1}y_i, \cdots, y_i^n)^T \tag{9c}$$

where $\mathbf{c}$ is the model parameter vector of dimension $M = n^3 + 3n/3$, and $\mathbf{f}_i$ the image coordinate polynomial function. We choose the inverse impact time model (9a) because the impact time appears in the estimation constraint by its inverse. Transformation of the terrain model from the inverse impact time format to normal impact time format, and from image coordinate representation (x_i, y_i) to the terrain coordinate representation (τ_{ix}, τ_{iy}) can be found in Negahdaripour & Lee [18].

Requiring that the terrain points (impact times) satisfy the geometry model (9a) means that (9a) can be substituted into (8). After substitution, the simplified snapshot estimation problem becomes one of determining the $\hat{\mathbf{\Omega}}$, $\hat{\mathbf{u}}_v$, and $\hat{\mathbf{c}}$ that minimize

$$r(\mathbf{\Omega}, \mathbf{u}_v, \mathbf{c}) \equiv \sum_{i=1}^{N} \left| I_{it} - \mathbf{h}_{i\Omega}^T\mathbf{\Omega} - \mathbf{c}^T\mathbf{f}_i\mathbf{h}_{iv}^T\mathbf{u}_v \right|^2 \tag{10}$$

across all feasible model parameters $\mathbf{c}$, angular velocity $\mathbf{\Omega}$, and unit-length heading vector $\mathbf{u}_v$. Without loss of generality, we can rewrite problem (10) as to determine the $\hat{\mathbf{\Omega}}$, $\hat{\mathbf{u}}_v$, and $\hat{\mathbf{c}}$ that minimize

$$r(\mathbf{\Omega}, \mathbf{v}, \mathbf{c}) = \sum_{i=1}^{N} \left| I_{it} - \mathbf{h}_{i\Omega}^T\mathbf{\Omega} - \mathbf{c}^T\mathbf{f}_i\mathbf{h}_{iv}^T\mathbf{v} \right|^2 \quad \text{subject to } |\mathbf{c}| = 1 \tag{11}$$

while allowing $\mathbf{v} \neq 1$.

The residual function $r(\mathbf{\Omega}, \mathbf{v}, \mathbf{c})$ of (11) is a bilinear function of motion states Ω, $\mathbf{v}$ and unit-length terrain parameter vector $\mathbf{c}$. In other words, if either $(\Omega, \mathbf{v})$ or $\mathbf{c}$ were known, estimation of $\mathbf{c}$ or $(\Omega, \mathbf{v})$ would be an easily solvable linear least squares (LS) minimization problem. We thus employ a two stage integration approach to estimate $(\Omega, \mathbf{v})$ and $\mathbf{c}$.

Specifically, given an estimate of the unit-length parameter vector $\mathbf{c}$, the snapshot terrain shape estimation problem of (11) becomes a linear least squares (LS) estimation problem to determine the $(\hat{\mathbf{\Omega}}, \hat{\mathbf{u}}_v)$ that minimize

$$r(\mathbf{\Omega}, \mathbf{v}) = \sum_{i=1}^{N} \left| I_{it} - \mathbf{h}_{i\Omega}^T\mathbf{\Omega} - (\hat{\mathbf{c}}^T\mathbf{f}_i\mathbf{h}_{iv}^T)\mathbf{v} \right|^2 \tag{12}$$

The snapshot estimates $(\hat{\mathbf{\Omega}}, \hat{\mathbf{v}})$ and their covariance's $\mathbf{P}_\Omega$ and $\mathbf{P}_v$ can thus be found using the conventional Singular Value Decomposition (SVD) LS algorithm [19].

The second stage assumes $(\hat{\mathbf{\Omega}}, \hat{\mathbf{v}})$ known so that the snapshot terrain shape estimation problem becomes a constrained linear LS estimation problem to minimize

$$r(\hat{\mathbf{c}}) = \sum_{i=1}^{N} \left| b_i - (\hat{\mathbf{v}}^T\mathbf{h}_{iv}\mathbf{f}_i^T)\mathbf{c} \right|^2 \text{ subject to } |\mathbf{c}| = 1 \tag{13a}$$

where

$$b_i \equiv I_{it} - \mathbf{h}_{i\Omega}\hat{\mathbf{\Omega}} \tag{13b}$$

The snapshot estimate $\hat{\mathbf{c}}$ and its covariance $\mathbf{P}_c$ can then be found using the constrained Singular Value Decomposition (SVD) LS algorithm given in appendix A.

Figure 4 summarizes this two stage integration approach. Starting with an initial guess on $\mathbf{c}$, the snapshot state estimator generates an estimate of Ω and $\mathbf{v}$, which in turn drive the snapshot terrain estimator to update the estimate on $\mathbf{c}$. The updated $\mathbf{c}$ is then fed back to the snapshot state estimator to generate a new estimate of the motion states. The iteration proceeds until a convergence criterion on residual function reduction is satisfied. After the convergence, the impact time map estimate $\hat{\tau}_{iz}$ and its variance $\sigma(\hat{\tau}_{iz})$ can then be determined via

$$\hat{\tau}_{iz} = |\mathbf{v}| / \hat{\mathbf{c}}^T\mathbf{f}_i \quad \text{and} \quad \sigma_{iz} = \mathbf{f}_i^T\mathbf{P}_c\mathbf{f}_i / \hat{\tau}_{iz}^2 \tag{14}$$

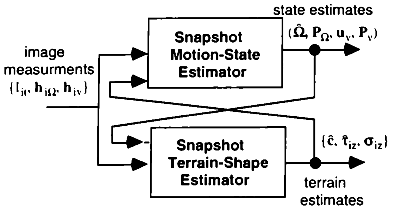

state estimates
$(\hat{\Omega}, P_\Omega, u_v, P_v)$
image
measurments
$\{I_{it}, h_{i\Omega}, h_{iv}\}$
Snapshot
Motion-State
Estimator
Snapshot
Terrain-Shape
Estimator
$\{\hat{c}, \hat{\tau}_{iz}, \sigma_{iz}\}$
terrain
estimates

Fig 4. Geometry Model Based Snapshot Estimator

Note that a pure guess on c is needed only for the first snapshot estimate. The snapshot estimate at any other later time (frame) can always use the value of the previous c as a starting estimate since the terrain geometry, in general, does not change much from one image frame to another.

State estimation complexity is of the order 36N flops, while terrain model estimation complexity is of the order 9N flops. If the snapshot estimator takes L iterations to converge, then overall computational complexity is of the order 45LN flops.

C. Kalman Filter

Using the snapshot state and terrain impact time estimates as measurements, we employ a motion state Kalman filter and an impact time map Kalman filter to incrementally improve the motion and terrain shape estimates. Figure 5 shows the architecture of the motion and terrain Kalman filter. Notice that the impact time map Kalman filter uses the smoothed motion state estimates as its inputs since we specify impact time evolution as a function of the observer motion as explained below.

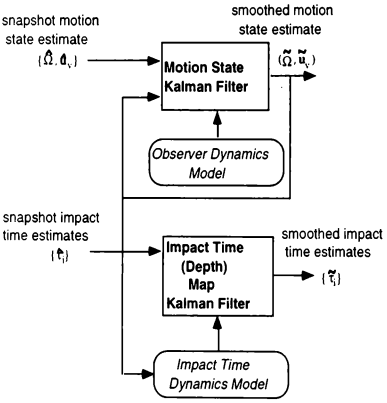


Fig 5. Kalman Filter

The Kalman filter is a Bayesian estimation technique for estimating stochastic dynamic system states using noisy observations or measurements. Implementation of a Kalman filter requires specification of three probabilistic models: the system model, the measurement model, and the prior model. The system model describes the evolution of system state over time. The measurement model relates the system state to the observed measurements. The prior model describes the initial knowledge about the system state and its variance before the first measurement is taken. Once the three models are specified, the Kalman filter can then be implemented using the filter algorithm given in appendix B.

Model complexity is primarily determined by the selection of a coordinate system: the proper system can often transform an apparently intractable filtering problem into one that is readily solvable. For this problem, we choose an observer-fixed coordinate system (i.e., the body coordinate system) since it leads to linear dynamics in both the motion and impact time system models.

We begin by deriving the dynamics equations for the observer's translational and rotational motions. Based on these dynamics equations, we then specify the system and measurement models for the motion state Kalman filter. The specification of the prior models is done implicitly by assuming zero knowledge about state variables before the first measurement.

Let k be the current time (frame) index and $\mathbf{R}_k$ represent the position of the observer at k. For a small frame time Δt, assuming constant translational velocity between frames, we have the translational dynamics equations

$$\mathbf{R}_k = \mathbf{R}_{k-1} + \mathbf{V}_{k-1}\Delta t$$

$$\mathbf{V}_k = \mathbf{V}_{k-1} \qquad (15a, b)$$

In our chosen observer-fixed coordinate system $\mathbf{R}_k = 0$ since the current position of the observer is always the origin of the coordinate system. Consequently, the selection of the observer-fixed coordinate system reduces the system dynamics equations to (15b).

Using a spherical coordinate representation, the dynamics of (15b) can also be written as

$$V_k = V_{k-1}$$

$$\psi_k = \psi_{k-1}$$

$$\gamma_k = \gamma_{k-1} \qquad (16a, b, \&c)$$

where $\mathbf{V}_k = V_k(\cos\psi_k \sin\gamma_k, \sin\psi_k \sin\gamma_k, \cos\gamma_k)^T$

The snapshot estimator provides measurements or observations only on the heading vector $\hat{\mathbf{u}}_{v,k}$ which is a unit vector defined by

$$\hat{\mathbf{u}}_{v,k} = (\cos\hat{\psi}_k \sin\hat{\gamma}_k, \sin\hat{\psi}_k \sin\hat{\gamma}_k, \cos\hat{\gamma}_k)^T \qquad (17)$$

where ψ_k is the observer's heading angle and γ_k is the observer's flight path angle γ_k. From the snapshot estimate $\hat{\mathbf{u}}_{v,k}$, the heading angle estimate $\hat{\psi}$ and the flight path angle estimate $\hat{\gamma}_k$ can be computed directly via

$$\hat{\psi}_k = \tan^{-1}\left(\frac{\hat{\mathbf{u}}_{v,k}(2)}{\hat{\mathbf{u}}_{v,k}(1)}\right)$$

$$\hat{\gamma}_k = \cos^{-1}(\hat{\mathbf{u}}_{v,k}(3)) \qquad (18a, b)$$

Consequently, we define a new state variable—heading angle vector $\mathbf{\Phi}_k = (\psi_k, \gamma_k)^T$ and specify the translational system and measurement models as

$$\mathbf{\Phi}_k = \mathbf{\Phi}_{k-1} + \mathbf{\mu}_{k-1}$$

$$\hat{\mathbf{\Phi}}_k = \mathbf{\Phi}_k + \mathbf{\eta}_k \qquad (19a, b)$$

where zero-mean Gaussian random variables $\mathbf{\mu}_k$ and $\mathbf{\eta}_k$ account for the unmodeled dynamics and observation errors, respectively.

We now derive the dynamics equations for the observer's rotational motion. There are many different ways to describe an observer's rotational motion (e.g. the nine-parameter expression, the quaternion expression, the Rodrigurcs expression, and the conventional Euler angle roll-pitch-yaw expression). Each expression leads to a different system model. We select the roll-pitch-yaw expression that uses three consecutive rotations around three principle axes to characterize an arbitrary rotation. Let $\boldsymbol{\theta} = (\theta_x, \theta_y, \theta_z)^T$ be the three Euler angles—yaw, pitch, and roll angles defining the rotations around the three axes (X, Y, Z) of a given coordinate system. The corresponding rotation matrix **T** is given as follows:

$$\mathbf{T} = \mathrm{RPY}(\boldsymbol{\theta}) \equiv \mathrm{rot}(\theta_z)\,\mathrm{rot}(\theta_y)\,\mathrm{rot}(\theta_x) \tag{20a}$$

where

$$\mathrm{rot}(\theta_x) = \begin{bmatrix} \cos\theta_x & \sin\theta_x & 0 \\ -\sin\theta_x & \cos\theta_x & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathrm{rot}(\theta_y) = \begin{bmatrix} \cos\theta_y & 0 & -\sin\theta_y \\ 0 & 1 & 0 \\ \sin\theta_y & 0 & \cos\theta_y \end{bmatrix}$$

$$\mathrm{rot}(\theta_z) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta_z & \sin\theta_z \\ 0 & -\sin\theta_z & \cos\theta_z \end{bmatrix} \tag{20b, c, \&d}$$

Let the matrix $\mathbf{T}_k$ represent the observer's current orientation specified using the roll-pitch-yaw expression. For a small frame time Δt, assuming constant rotation angles $\boldsymbol{\theta}_k$ between frames, we have the following rotation dynamics equations:

$$\mathbf{T}_k = \mathbf{T}_{k-1}\,\mathrm{RPY}(\boldsymbol{\theta}_{k-1})$$

$$\mathrm{RPY}(\boldsymbol{\theta}_k) = \mathrm{RPY}(\boldsymbol{\theta}_{k-1}) \tag{21a, b}$$

Since we have chosen an observer-fixed coordinate system, $\mathbf{T}_k = \mathbf{I}$ since the current orientation of the observer defines the observer's body coordination system. Consequently, the selection of the observer-fixed coordinate system reduces the dynamics equations to (21b).

Although the dynamics equation (21b) completely define the rotational motion, it is not a system model since the nine-element rotation matrix $\mathrm{RPY}(\boldsymbol{\theta}_k)$ is uniquely specified by the three- variable rotation angles

$$\boldsymbol{\theta}_k = \boldsymbol{\Omega}_k \Delta t \tag{22}$$

Let RPY^{-1} be the inverse operator of RPY. That is, RPY^{-1} takes a rotation matrix as the operand and returns its Euler angles. Applying RPY^{-1} on both sides of (21b) and canceling the constant Δt, we have the rotation system model, given by

$$\boldsymbol{\Omega}_k = \boldsymbol{\Omega}_{k-1} + \boldsymbol{\varepsilon}_{k-1} \tag{23a}$$

where zero-mean Gaussian random variables $\boldsymbol{\varepsilon}_k$ accounts for the unmodeled dynamics. We specify the rotation measurement model as

$$\hat{\boldsymbol{\Omega}}_k = \boldsymbol{\Omega}_k + \boldsymbol{\zeta}_k \tag{23b}$$

where zero-mean Gaussian random variables $\boldsymbol{\zeta}_k$ account for the observation errors.

Using the snapshot angular velocity and heading angle estimates as the measurements, the translational and rotational system models of (19a) and (23a), and measurement models of (19b) and (23b), we can then directly use the Kalman filter algorithm to generate the smoothed motion state estimates.

The impact time map Kalman filter generates the smoothed impact time estimates using the snapshot estimates of the impact time as measurements. We adopt a pixel-based (iconic) approach to the Kalman filter implementation, where the impact time at a pixel is defined as the state variable.

Given that we have chosen the body coordinate system for the system dynamics description, and assuming a rigid terrain surface, the system model for the ith pixel impact time (state variable) can be expressed as

$$\tau_{i,k} = \tau_{i,k-1} + \varepsilon_{k-1} \tag{24}$$

where both $\tau_{i,k}$ and $\tau_{i,k-1}$ are represented in the body coordinate system at time (frame) k, and ε_{k-1} is a zero-mean Gaussian noise accounting for the model error. The measurement model is given by

$$\hat{\tau}_{i,k} = \tau_{i,k} + \xi_k \tag{25}$$

where ξ_k represents the measurement error.

The snapshot estimator generates the measurement $\hat{\tau}_{i,k}$ and its variance $\sigma_{i,k}$ at each pixel. Given the system and measurement models, the implementation of the Kalman filter for impact time smoothing is straight forward except for computation of the estimate $\hat{\tau}^-_{i,k}$ in the prediction phase according to (B5). The available estimate $\hat{\tau}_{i,k-1}$ is made in the body coordinate system at time k-1 while the computation of $\hat{\tau}^-_{i,k}$ needs the estimate made in the body coordinate system at time k.

We develop a numerical extrapolation approach to generating the needed estimate of $\hat{\tau}^-_{i,k}$ at time k, using the available impact time map estimates $\{\hat{\tau}_{i,k-1}\}$ at k-1 and the known observer motion states $\hat{\boldsymbol{\Omega}}_{k-1}$ and $\hat{\mathbf{u}}_{v,k-1}$. Figure 6 illustrates the impact time evolution between two consecutive frames. The evolution is determined by two key factors: relative motion of the observer with respect to the terrain and a finite field-of-view (FOV) limiting the observer's instantaneous view of the terrain. Specifically, the relative motion implies that $\hat{\tau}_{i,k}$ will change from one image frame to another since it is measured relative to the observer. The limited FOV implies that some new terrain (thus its impact times) will enter the impact time map and some old terrain will move out of the map.

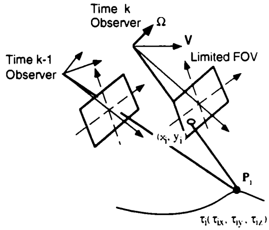


Fig. 6. Observer Motion Changes

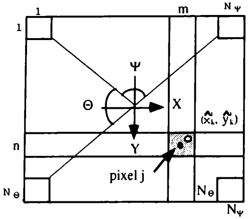


Fig. 7. Image Coordinates

At time k-1, the estimated impact time vector of the ith terrain point is given by

$$\hat{\tau}_{i,k-1} = (\hat{\tau}_{iz,k-1}x_i, \hat{\tau}_{iz,k-1}y_i, \hat{\tau}_{iz,k-1})^T \tag{26}$$

from the perspective projection. At time k, the observer and his associated coordinate system move to a new location. In the observer-fixed coordinate system, the orientation change due to rotations can be determined from (21a) and is given by

$$\mathbf{T}_{k-1} = [RPY(\hat{\mathbf{\Omega}}_{k-1}\Delta t)]^T \tag{27}$$

The impact time change due to translation can be determined from (15a) by normalizing the equation using the observer speed V_{k-1}

$$\tau_{k-1} = -\hat{\mathbf{u}}_{v,k-1}\Delta t \tag{28}$$

Given the estimated impact time vector $\hat{\boldsymbol{\tau}}_{i,k-1}$, the angular velocity estimate $\hat{\mathbf{\Omega}}_{k-1}$, and the heading vector estimate $\hat{\mathbf{u}}_{v,k-1}$, the impact time vector at time k can be estimated as

$$\hat{\boldsymbol{\tau}}_k^i = [RPY(\hat{\mathbf{\Omega}}_{k-1}\Delta t)]^T[\hat{\boldsymbol{\tau}}_{i,k-1} - \hat{\mathbf{u}}_{v,k-1}\Delta t] \tag{29}$$

where the superscript i denotes that $\hat{\boldsymbol{\tau}}_k^i$ is computed using the impact time vector $\hat{\boldsymbol{\tau}}_{i,k-1}$ of pixel i at time k-1. Notice that because of the observer motion and the limited horizon, the estimated impact time vector $\hat{\boldsymbol{\tau}}_k^i$ would not project to pixel i at k, but to new image coordinates $(\hat{x}_k^i, \hat{y}_k^i)^T$, determined by

$$(\hat{x}_k^i, \hat{y}_k^i)^T = (\frac{\hat{\tau}_{x,k}^i}{\hat{\tau}_{z,k}^i}, \frac{\hat{\tau}_{y,k}^i}{\hat{\tau}_{z,k}^i})^T \text{ for } i = 1, 2, ..., N \tag{30}$$

As shown in figure 7, the closest pixel row index $\hat{m}$ and column index $\hat{n}$ for the image coordinates $(\hat{x}_k^i, \hat{y}_k^i)^T$ can be computed, respectively, by

$$\hat{m} = int[(\frac{\hat{x}_k^i N_\Psi}{\tan(\Psi/2)} + N_\Psi + 1)/2] \tag{31}$$

$$\hat{n} = int[(\frac{\hat{y}_k^i N_\Theta}{\tan(\Theta/2)} + N_\Theta + 1)/2] \tag{32}$$

where *int*(d) is an integer operator which returns an integer that is closest to the value d, Θ is the vertical FOV angle, N_Θ is the number of the vertical pixels, Ψ is the horizontal FOV angle, and N_Ψ is the number of horizontal pixels.

As shown in figure 7, an image point $(\hat{x}_k^i, \hat{y}_k^i)^T$ either stays within the image FOV at k if $1 \le \hat{m} \le N_Y$ and $1 \le \hat{n} \le N_Q$, or moves out if one of the inequality constraints does not hold. In the first case, we consider that pixel $j = (\hat{n} - 1)N_\Psi + \hat{m}$ at k has an impact time $\hat{\tau}_k^i$ and a variance $P_{j,k}^-$, computed in the prediction phase of the Kalman filter by

$$\hat{\tau}_{j,k}^- = \hat{\tau}_k^i \tag{33a}$$

$$P_{j,k}^- = P_{i,k} / \alpha \tag{33b}$$

where $\alpha \le 1$ is a discount factor accounting for the uncertainty in impact time estimation via numerical extrapolation.

IV. PERFORMANCE EVALUATION

This section evaluates overall operation of the motion and terrain shape estimator. We first define the performance evaluation criterion. Next, we present evaluation results using computer generated flat terrain imagery. We then present the evaluation results using flight-recorded imagery.

A. Performance Criterion

To evaluate system performance for both the computer generated and real images, several performance indices are defined. These include:

- *Aimpoint Error:* This is obtained from the angular difference between true heading $\mathbf{u}_V$ and heading $\hat{\mathbf{u}}_v$ estimated by the estimator. The aimpoint error reflects total aim estimation error in both vertical and lateral directions, and is computed by

$$\varepsilon_{aim} = \cos^{-1}(\mathbf{u}_v \bullet \hat{\mathbf{u}}_v) \tag{34}$$

- *Angular Rate Error:* This is obtained from the magnitude of the difference between actual angular velocity $\mathbf{\Omega}$ and angular velocity $\hat{\mathbf{\Omega}}$ estimated by the estimator. The angular rate error reflects total body-rate error across all three body axes, and is computed by

$$\varepsilon_{arg} = |\mathbf{\Omega} - \hat{\mathbf{\Omega}}| \tag{35}$$

- *Impact Time Error:* This is obtained by first determining the difference between the true pixel-dependent terrain-surface impact time τ_i with the corresponding estimate $\hat{\tau}_i$ generated by the estimator, at each visible ground pixel, and then averaging the results. Note that when the impact time is computed in the navigation system, or in the body coordinate system of an observer undergoing straight and level flight, the average impact time equals the observer's altitude normalized by his speed. Assuming that the system has access to an accurate estimate of the observer speed v, this error index reflects the precision of the system in estimating the effective altitude (range) of the observer above the terrain, in time units.
- *Across-Frame Root-Mean Square (RMS) Error:* This is obtained by first computing the error mean m and standard deviation s across multiple frames. The RMS error is then computed via

$$\text{RMS} = \sqrt{\mu^2 + s^2} \tag{36}$$

B. Estimation using Computer Generated Imagery

A sequence of images was computer generated using a flat-terrain image generator. The simulated vehicle flies straight-and-level with a speed of 300 ft/sec at an altitude of 200 feet above the terrain, with zero body attitude rates. The vehicle sensor was boresighted 15° below the horizon. The sensor field-of-view (FOV) was set to 32° laterally by 32° vertically, and the pixel count was 32 laterally by 32 vertically to yield a nominal 1° by 1° pixel. The simulated terrain is flat and featureless and is textured via a spatial 2D sinusoidal pattern with a wavelength of 500 feet in both vertical and lateral directions.

Figure 8a and 8b show one image frame generated under the above conditions. With a resolution of 128 x 128 pixels, figure 8a approximates the true unpixellated terrain image seen through the sensor field-of-view. The upper 10% of the figure is sky, and has a neutral gray value. The lower portion below the horizon shows the terrain pattern, which naturally recedes to the horizon. Note the regular 2D sinusoidal pattern used to "decorate" the surface. Figure

8b illustrates the image actually *imaged* by the sensor at the 32 x 32 pixel resolution. The effects of pixellation are clearly evident.

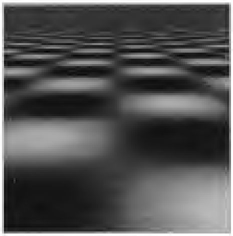


Fig. 8a. High Resolution Image

Fig. 8.b Low Resolution Image

In the simulation, a sequence of consecutive images like figure 8b are generated, and then processed, frame by frame, by the motion and terrain shape estimator to obtain the state and terrain shape estimates. For the simulations described in this section, the interframe sample time is set at 0.02 sec (a frame rate of 50 Hz), so that an imaged terrain point in the center of the sensor focal plane will move at about one pixel/frame, at the given nominal altitude, speed, and boresight angle.

The estimator used a flat terrain model, and assumed an initial +5° error in both the flight path and the heading angles to reflect the initial uncertainty in the observer motion state. The model standard deviations for heading and path angles were set at 0.01°, the model variances for angular velocity vector were set at 0.01 °/sec^2 for each individual vector element. The discount factor for terrain impact time (range) map evolution was set at 0.9.

Figure 9 shows a time history for the observer impact time (altitude/speed) estimate, which is the cross-pixel average of the impact time map in the body coordinate system. In the figures, the solid line represents the snapshot estimate, the thick solid line represents the Kalman filter smoothed estimate, and the dashed line represents the true observer state and terrain shape parameters. As shown, both the snapshot and filtered impact times oscillate around the actual impact time. Note that a larger variation occurs in the snapshot estimate; the effects of the Kalman filter in smoothing the oscillation of the snapshot estimate and improving estimation accuracy are clearly demonstrated. Oscillation in the estimate is due to the periodic nature of the sinusoidal pattern being viewed.

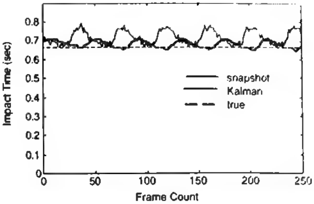


Fig. 9. Impact Time Estimation History

Figures 10 to 12 illustrate error histories for the snapshot and filtered estimates of motion state and impact time. As shown in figure 10, the snapshot impact time estimate has an oscillating error of 0% to 15%, which is reduced to within a 5% error zone after Kalman filtering.

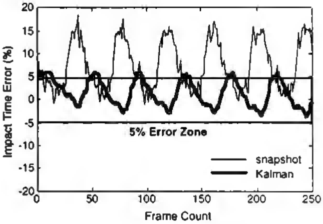


Fig. 10. Impact Time Error History

Figure 11 illustrates the error histories of the snapshot and filtered aimpoint estimates. The snapshot estimation error oscillates between 2° to 4°, while the filtered estimation error converges to about 1.2°.

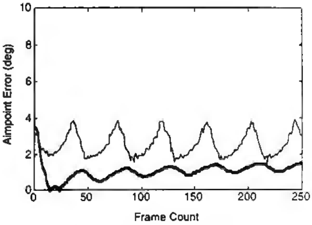


Fig. 11. Aimpoint Error History

Figure 12 illustrates the error histories of the snapshot and filtered angular rate estimates. The snapshot estimation error shows an oscillating error between 1.5 to 4°, while the filtered estimation error shows a slightly oscillating 3° error.

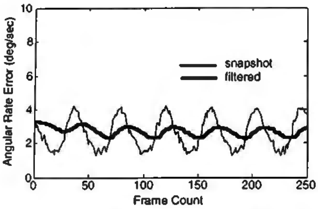


Fig. 12. Angular Rate Error History

RMS errors, using the geometry based estimator, in the filtered impact time, aimpoint, and angular rate are 3.02%, 1.09°, and 2.77°/sec, respectively.

C. Evaluation Using Real Imagery

Evaluation using real imagery was done using a flight-recorded and real-time frame grabbed terrain image sequence obtained from the NASA Ames Research Center. The image sequence was acquired by a camera mounted under a helicopter nose and oriented roughly in the direction of flight. The position of the camera and its orientation with respect to the helicopter were held constant throughout the flight. The methodology used to generate the image sequence and truth data are described in Smith [21]. The helicopter was flying with a speed of approximately 20 knots at an altitude of 15 ft above a runway. The images were captured at 30 Hz, so that the helicopter moved approximately 1 ft between successive images. Figures 13a and 13b show the first and the last images of a sequence of 90 images. Each image has a resolution of 442 x 512 pixels at 256 gray levels.

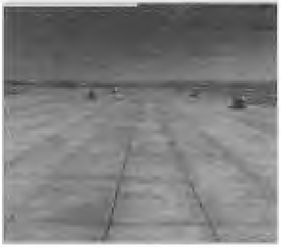


Fig. 13a. First Image of NASA Sequence (High Resolution)

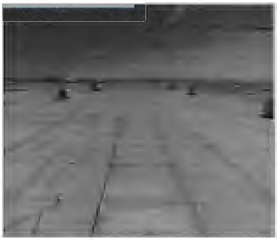


Fig. 13b. Last Image of NASA Sequence (High Resolution)

We first used the images at their original 442 x 512 pixel resolution without any pre-processing of images. For the snapshot estimator, a flat terrain model was adopted. For the Kalman filter, the model standard deviations for heading and flight path angle dynamics were set at 0.01°, and the model standard deviations for angular velocity dynamics were set at 0.01 °/sec^2, equally for the three angular velocity components. The discount factor for terrain impact time (range) map evolution dynamics was set as 0.9. In processing the images, the pixels above the horizon were not used.

Figures 14 to 16 show time histories for the estimated helicopter impact time (altitude/speed), heading angle, flight path angle, and three individual angular velocity components. The plots are drawn by plotting the estimate and measured state and terrain parameters against frame count with a frame interval time of 1/30 sec. In the figures, the solid line represents the snapshot estimates, the thick solid line represents the Kalman filter smoothed estimates, and the dashed line represents the helicopter state and terrain parameters measured using on-board navigation systems.

As shown in figure 14, the impact time estimate has quite a large error. Two factors contribute to this. The first is the lack of intensity contrast in the "non-decorated" real imagery, as demonstrated in figure 13. The second is the small pixel size that leads to a high noise/signal ratio in the image gradient computation. These two factors, however, seem not to affect the motion state estimation accuracy adversely. Figures 15 and 16 illustrate the time histories of the estimated heading and flight path angles, respectively. The ability of the estimator to track the helicopter motion state dynamics is clearly demonstrated in plots of the three angular velocity components, shown in figures 16a, 16b, and 16c, respectively. Note how closely and swiftly each estimated angular velocity component follows the dynamics of its corresponding true angular velocity component.

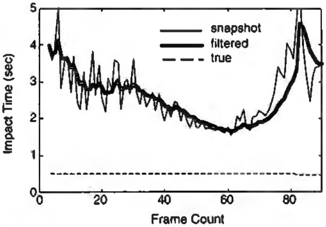


Fig. 14. Impact Time Estimation History (High Resolution)

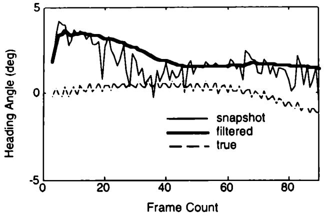


Fig. 15a. Heading Angle Estimation History (High Resolution)

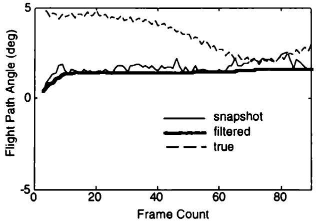


Fig. 15b. Flight Path Angle Estimation History (High Resolution)

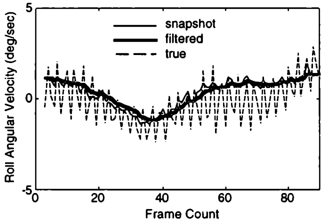


Fig. 16a. Roll Angular Velocity Estimation History (High Resolution)

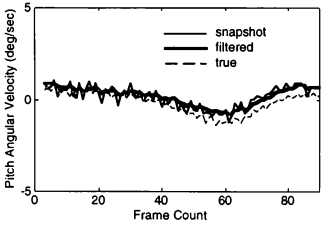


Fig. 16b. Pitch Angular Velocity Estimation History (High Resolution)

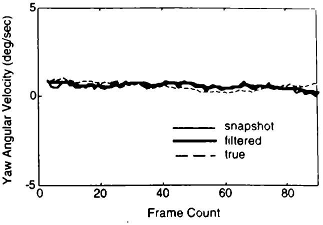


Fig. 16c. Yaw Angular Velocity Estimation History (High Resolution)

Figures 17 and 18 show time histories of the aimpoint error and angular rate error, respectively. After the first 30 frames (1 sec), the aimpoint error stays below 2.5°, and the angular rate estimate error stays below 2°/sec. The RMS errors for aimpoint and angular rate estimates are 2.40° and 1.29 °/sec, respectively.

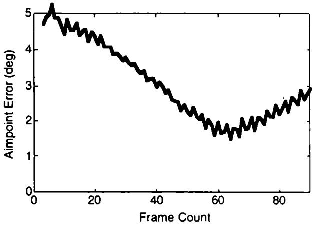


Fig. 17. Aimpoint Error History (High Resolution)

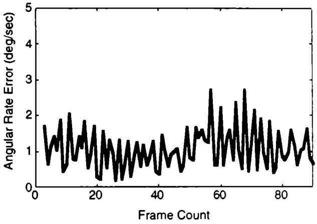


Fig. 18. Angular Rate Error History (High Resolution)

Estimation using high resolution images (442x512) leads to a rather large impact time errors. Since the model based estimator is best suited for simple geometry terrain without objects and uses only low-frequency terrain motion, superpixellation of the original images is called for. Figures 19a and 19b show the superpixellated versions of the images illustrated in 13a and 13b, at a resolution of 56 x 64 pixels, using 8 x 8 kernels. The trucks are no longer recognizable, but the low frequency imagery of the runway is.

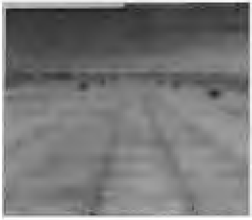


Fig. 19a: First Image of NASA Sequence (Low Resolution)

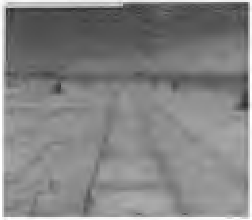


Fig. 19b. Last Image of NASA Sequence (Low Resolution)

Figures 20 to 22 show the estimation time histories for the helicopter impact time (altitude/speed), heading angle, flight path angle, and three individual angular velocity components, respectively. As shown in figure 20, the impact time estimate now approaches the true impact time. The difference between the true and the filtered impact time is about 0.05 sec or 1.6 ft. This shows that for overall terrain surface estimation superpixellation does not adversely affect estimation accuracy but enhances it.

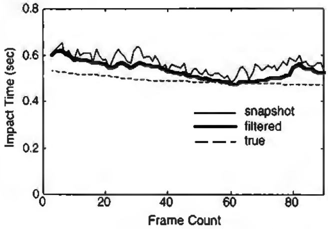


Fig. 20. Impact Time Estimation History (Low Resolution)

Figures 21a and 21b illustrate the time histories of the heading and flight path angle estimates, respectively. Superpixellation does not affect the heading angle estimation; but seems to introduce a consistent negative flight path angle bias to the flight path estimation. Note, however, how the system estimates rapidly follow the trend of the true helicopter motion state states.

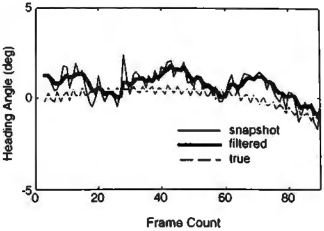


Fig. 21a: Heading Angle Estimation History (Low Resolution)

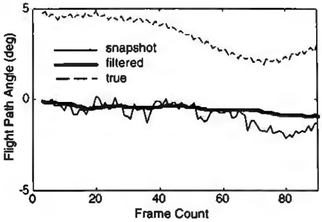


Fig. 21b: Flight Path Angle Estimation History (Low Resolution)

Angular velocity estimation is not adversely affected by superpixellation, as shown in figures 22a, b, and c. The three estimated angular velocity components track and follow the dynamics of the true angular velocity closely.

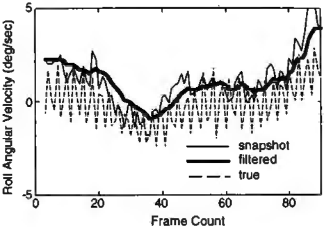


Fig. 22a. Roll Angular Velocity Estimation History (Low Resolution)

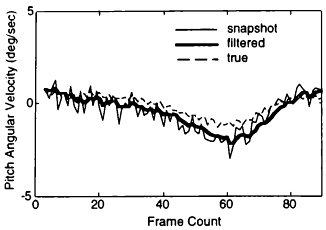


Fig. 22b. Pitch Angular Velocity Estimation History (Low Resolution)

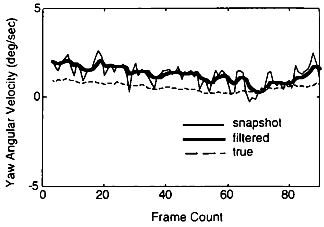


Fig. 22c. Yaw Angular Velocity Estimation History (Low Resolution)

Figures 23, 24, and 25 show the time histories of the impact time error, aimpoint error and angular rate error, respectively. After the first 30 frames (1 sec), the impact time error almost stays within a 10% error zone, the aimpoint error stays below 3°, and the angular rate error stays below 2°/sec. The RMS errors for the impact time, aimpoint, and angular rate estimates are 7.60%, 3.52°, and 1.93 °/sec, respectively.

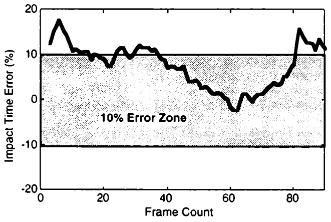


Fig. 23. Impact Time Error History (Low Resolution)

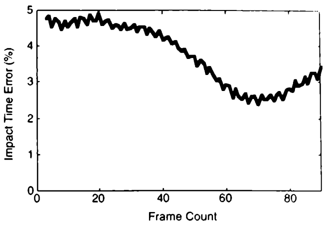


Fig. 24. Aimpoint Error History (Low Resolution)

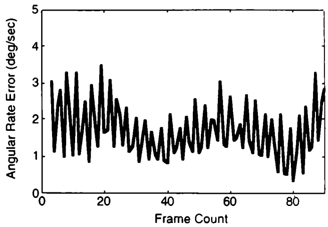


Fig. 25. Angular Rate Error History (Low Resolution)

V. SUMMARY

This paper develops, implements, and evaluates a systematic and practical solution to passive navigation. Three tasks were involved.

We first reviewed current approaches and evaluated their advantages and disadvantages. The review results indicated that although there exist many solutions to passive navigation, from a practitioner's point of view, the problem was still mostly unsolved because of various limitations and complexities of published solutions.

We next developed and implemented a systematic and practical solution to passive navigation. A three step passive vision motion and terrain shape estimator was developed consisting of: 1) an image processor; 2) a snapshot motion and terrain shape estimator; and 3) a Kalman filter to smooth the snapshot estimates. This modular estimator provided performance improvements and simplification for system implementation over other methods.

Following implementation of the estimator, we evaluated system performance via an extensive testing program. The estimator was tested using both CGI and real imagery. The evaluation results indicated that the estimator is capable of accurate, efficient, and robust motion and terrain shape estimation. For the CGI, the estimation errors were less than 2° in aimpoint, less than 3°/second in angular rate, and less than 5% in altitude (impact time). For the flight-recorded and real-time grabbed terrain images, even though they were poorly "decorated", had very high noise-to-signal ratios, and the observer was not undergoing constant motion, the estimator was still capable of generating accurate motion and terrain shape estimates. Specifically, it demonstrated errors less than 5° in aimpoint, less than 2°/second in angular rate, and less than 10% in altitude (impact time). Moreover, the estimator showed an excellent ability in rapidly following and tracking the non-constant observer motion.

The motion and terrain shape estimator presented in this paper bears some resemblance to earlier approaches in that we are attempting to avoid image flow computation and to enhance estimation accuracy. The current work, however, also makes a number of novel contributions. First, this work provides a systematic and practical solution to the problem, while previous studies usually are limited to restricted cases and require numerically intensive and computationally unstable solution algorithms. Second, the three-stage modular approach presented here removes the complicated interactions between low-level image processing and high-level motion state and terrain shape estimation in an integrated approach, and has a natural interface to incorporate external information for performance enhancement. For example, if the external information on observer motion state is available, it can then be easily incorporated into the Kalman filter without reconfiguration of the entire system. The separation of image processing, snapshot estimation, and Kalman filtering makes the solution to the overall motion and terrain estimation problem simpler, easier, and more robust. Third, a novel Kalman filter uses the snapshot motion and terrain estimates as measurements. The filter has linear models of both the motion and terrain, and can be easily implemented, in contrast to the extremely complicated extended non-linear Kalman filters proposed in other studies. Fourth, the estimator needs specification of only a few meaningful parameters since each precedent module is capable of generating almost all the information required by the subsequent module. Last but not least, the implementation of the estimator requires only conventional estimation algorithms (singular value decomposition least squares algorithm and linear system Kalman filter). The estimator has been tested on real terrain imagery and promising results have been achieved.

APPENDIX A: CONSTRAINED LEAST SQUARES ESTIMATION

The constrained linear LS estimation determines the estimate $\hat{\mathbf{z}}$ that minimizes

$$r(\mathbf{z}) \equiv |\mathbf{b} - \mathbf{A}\mathbf{z}|^2 \qquad \text{(A1a)}$$

subject to the constraint that

$$\mathbf{z}^T\mathbf{z} = 1 \qquad \text{(A1b)}$$

Using the Singular Value Decomposition algorithm [19], we can express matrix $\mathbf{A}$ as the product of an $N \times M$ column-orthogonal matrix $\mathbf{U}$, an $M \times M$ diagonal matrix $\mathbf{W}$ with positive or zero elements (the singular values), and the transpose of an $M \times M$ orthogonal matrix $\mathbf{V}$:

$$\mathbf{A} = \mathbf{U}\mathbf{W}\mathbf{V}^T \qquad \text{(A2)}$$

where

$$\mathbf{U}^T\mathbf{U} = \mathbf{V}^T\mathbf{V} = \mathbf{V}\mathbf{V}^T = \mathbf{I}, \text{ and } \mathbf{W} = \text{diag}\,(w_1, \cdots, w_M) \qquad \text{(A2b)}$$

Specifically, denoting

$$\mathbf{y} \equiv \mathbf{V}^T\mathbf{z} \quad \text{and} \quad \mathbf{c} \equiv \mathbf{U}^T\mathbf{b} \qquad \text{(A3)}$$

we have the LS problem (A1) being transformed to determine $\hat{\mathbf{y}}$ that minimizes

$$r(\mathbf{y}) = |\mathbf{c} - \mathbf{W}\mathbf{y}|^2 \qquad \text{(A4a)}$$

subject to the constraint that

$$\mathbf{y}^T\mathbf{y} = 1 \qquad \text{(A4b)}$$

Defining the Lagrange function

$$r(\mathbf{y}, \lambda) = |\mathbf{c} - \mathbf{W}\mathbf{y}|^2 + \lambda(\mathbf{y}^T\mathbf{y} - 1) \qquad \text{(A5)}$$

and noting that $\mathbf{W}$ is a diagonal matrix, we have the parametric solution $\mathbf{y}(\lambda)$ to (A4) given by

$$y_i = \frac{w_i c_i}{w_i^2 + \lambda}, \qquad \text{for } i = 1, 2, \ldots, M \qquad \text{(A6)}$$

Substituting (A6) into $\mathbf{y}^T\mathbf{y} = 1$, the Lagrange multiplier λ, is then constrained by the following single variable equation:

$$\sum_{i=1}^{M} \left(\frac{w_i c_i}{w_i^2 + \lambda}\right)^2 = 1 \qquad \text{(A7)}$$

The equation has a unique solution and can be found using standard root-finding algorithms, such as Newton' s method given in Press, Teukolsky, Vetterling & Flannery [19]. After the Lagrange multiplier λ is found, the original parameter vector $\hat{\mathbf{z}}$ and its estimated error covariance can be found via

$$\hat{\mathbf{z}} = \mathbf{V}\hat{\mathbf{y}} \qquad \text{where } \hat{\mathbf{y}} = \left(\frac{w_1 c_1}{w_1^2 + \lambda}, \cdots, \frac{w_M c_M}{w_M^2 + \lambda}\right)^T \qquad \text{(A8)}$$

$$\text{Cov}(\hat{\mathbf{z}}) = \mathbf{V}\,\text{diag}\left(\frac{1}{w_1^2 + \lambda}, \cdots, \frac{1}{w_M^2 + \lambda}\right)\mathbf{V}^T \qquad \text{(A9)}$$

APPENDIX B: KALMAN FILTER

The Kalman filtering approach is a Bayesian estimation technique used to obtain accurate and robust estimates of stochastic dynamic system states that are observed with noisy measurements.[16] The implementation of a Kalman filter requires specification of three probabilistic models: system, measurement, and prior estimate.

The *system model* describes the system state evolution over time, and is defined by the state-space transition equation

$$\mathbf{x}_k = \mathbf{A}_{k-1}\mathbf{x}_{k-1} + \boldsymbol{\varepsilon}_k, \qquad \boldsymbol{\varepsilon}_k \sim N(0, Q_k) \qquad \text{(B1)}$$

where $\mathbf{x}_k$ is the current system state vector at time k, $\mathbf{A}_{k-1}$ is a known transition matrix, and $\boldsymbol{\varepsilon}_k$ is a Gaussian noise with zero mean and a covariance $\mathbf{Q}_k$.

The *measurement model* relates the state vector $\mathbf{x}_k$ to the noisy measurement vector $\mathbf{y}_k$ via a measurement matrix $\mathbf{C}_k$ and the addition of a Gaussian noise $\boldsymbol{\eta}_k$ with zero mean and a covariance $\mathbf{R}_k$. It is defined by the measurement equation

$$\mathbf{y}_k = \mathbf{C}_k\mathbf{x}_k + \boldsymbol{\eta}_k, \qquad \boldsymbol{\eta}_k \sim N(0, \mathbf{R}_k) \qquad \text{(B2)}$$

The prior estimate model describes the knowledge about the initial system state $\hat{\mathbf{x}}_0$ and its covariance $\mathbf{P}_0$ before the first measurement is taken, and assumes uncorrelated system and measurement noises. It is defined by

$$\hat{\mathbf{x}}_0 = E[\mathbf{x}_0] \qquad \text{(B3a)}$$

$$\mathbf{P}_0 = \text{CovE}[\mathbf{x}_0] \quad \text{and} \quad E[\boldsymbol{\varepsilon}_k\boldsymbol{\eta}_k^T] = 0 \qquad \text{(B3b \& c)}$$

We always assume that there is no initial knowledge on the system states before the first estimate. In other words, the prior model for the system states is fixed, and is always given by

$$\hat{\mathbf{x}}_0 = 0, \mathbf{P}_0 = \text{diag}\,(\infty) \qquad \text{(B4)}$$

Once the system, measurement, and prior models are specified, the Kalman filter algorithm operates in two phases, prediction and correction, as shown in figure B1 via the right and left blocks separated by a dashed line.

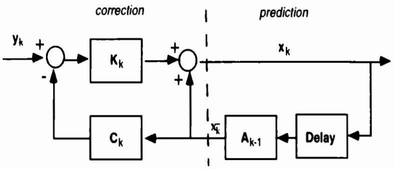


Fig. B1. Kalman Filter Block Diagram

In the prediction phase, the previous state estimate $\hat{\mathbf{x}}_{k-1}$ and covariance estimate $\mathbf{P}_{k-1}$ are extrapolated to predict the current state $\mathbf{x}_k^-$ and $\mathbf{P}_k^-$ via state extrapolation and covariance extrapolation defined by

$$\hat{\mathbf{x}}_k^- = \mathbf{A}_{k-1}\hat{\mathbf{x}}_{k-1} \tag{B5}$$

$$\mathbf{P}_k^- = \mathbf{A}_{k-1}\mathbf{P}_{k-1}\mathbf{A}_{k-1}^T + \mathbf{Q}_{k-1} \tag{B6}$$

In the correction phase, the predicted covariance is used to compute the Kalman gain matrix $\mathbf{K}_k$ and to update the covariance matrix $\mathbf{P}_k$ via

$$\mathbf{K}_k = \mathbf{P}_k^-\mathbf{C}_k^T[\mathbf{C}_k\mathbf{P}_k^-\mathbf{C}_k^T + \mathbf{R}_k]^{-1} \tag{B7}$$

$$\mathbf{P}_k = [\mathbf{I} - \mathbf{K}_k^-\mathbf{C}_k]\mathbf{P}_k^- \tag{B8}$$

The measurement residual $\mathbf{y}_k - \mathbf{C}_k\hat{\mathbf{x}}_k^-$ is then weighted by the gain matrix $\mathbf{K}_k$ and added to the predicted state $\mathbf{x}_k^-$ to yield the updated state estimate $\hat{\mathbf{x}}_k$ via

$$\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_k^- + \mathbf{K}_k[\mathbf{y}_k - \mathbf{C}_k\hat{\mathbf{x}}_k^-] \tag{B9}$$

The conventional Kalman filter algorithm is defined by equations (B5) to (B9). However, it is prone to serious numerical difficulties that can cause a loss of positive definiteness in updating the covariance matrix $\mathbf{P}_k$ leading to divergent state estimates. Although algebraically equivalent to the conventional Kalman filter algorithm, square-root Kalman filter algorithms exhibit improved numerical precision and stability, especially in those ill-conditioned problems where conventional algorithms fail.

Square root Kalman filter algorithms are developed using a square root decomposition feature of symmetric, positive semidefinite matrices—the class of matrices which the covariance matrix $\mathbf{P}_k$ belongs. This feature dictates that for each $\mathbf{P}_k$ there exists at least one square root matrix $\sqrt{\mathbf{P}_k}$ such that

$$\mathbf{P}_k = \sqrt{\mathbf{P}_k}\sqrt{\mathbf{P}_k}^T \tag{B10}$$

The basic idea of the square root algorithms is to replace the recursive updates of $\mathbf{P}_k$ in both the prediction and correction phases with the updates of $\sqrt{\mathbf{P}_k}$, and to update the state estimate using an optimal gain computed using $\sqrt{\mathbf{P}_k}$ instead of $\mathbf{P}_k$. The update of $\sqrt{\mathbf{P}_k}$ guarantees a positive definite covariance matrix $\mathbf{P}_k$, thus eliminating the possibility of state estimate divergence. Various versions of square root Kalman filter algorithm implementations are given in Maybeck [16].

ACKNOWLEDGMENTS

This work was performed under USAF Contract F41624-92-C-6007. Dr. Rik Warren of USAF Armstrong Laboratory was the Technical Monitor. We thank Dr. Ed Riley of Charles River Analytics for his technical assistance in both the completion of this contract and the creation of this document, and Ms. Kristy Shriver, Ms. April Blumenstiel, and Ms. Johanna Brown for their help in the creation and editing of this document. The authors also thank Mr. Phillip N. Smith and Mr. C. Thomas Synder of NASA Ames Research Center for letting us share their image data base.

REFERENCES

[1] P. Anandan, "A Computational Framework and an Algorithm for the Measurement of Visual Motion". *International Journal of Computer Vision*, vol 2, pp. 283-310, 1989.

[2] J. L. Barron, D. J. Fleet, and S. S. Beauchemin, "Systems and Experiment, Performance of Optical Flow Techniques". *International Journal of Computer Vision*, vol. 12, pp. 43-77, 1994.

[3] J. L. Barron, A. D. Jepson, and J. K. Tsotsos, "The Feasibility of Motion and Structure from Noisy Time-Varying Image Velocity Information". *Int. Journal of Computer Vision*, vol. 5, pp. 239-269, 1990.

[4] T. J. Broida, and R. Chellappa, "Estimation of Object Motion Parameters from Noisy Images". *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 8, pp. 90-99, 1986.

[5] D. J. Fleet, *Measurement of Image Velocity*. Norwell, MA: Kluwer Academic Publishers, 1992.

[6] D. J. Fleet and A. D. Jepson, "Computation of Component Image Velocity from Local Phase Information". *International Journal of Computer Vision*, vol. 5, pp. 77-104, 1990.

[7] D. J. Heeger, "Model for the Extraction of Image Flow". *Journal of the Optical Society of American A*, vol. 4, pp. 1455-1471, 1987.

[8] D. J. Heeger and A. D. Jepson, "Subspace Methods for Recovering Ridgid Motion I: Algorithm and Implementation". *International Journal of Computer Vision*, vol. 7, no. 2 pp. 95-117, 1992.

[9] J. Heel and S. Negahdaripour, "Time-Sequential Structure and Motion Estimation Without Optical Flow," *Proceeding of SPIE Symposium of Electronic Imag. Sens. Recons. 3D Objects and Scenes*, 1990.

[10] B. K. P. Horn and B. G. Schunck, "Determining Optical Flow". *Artificial Intelligence*, vol. 17, pp. 185-204, 1981.

[11] B. K. P. Horn & E. J. Weldon JR, "Direct Methods for Recovering Motion". *International Journal of Computer Vision*, vol. 2, pp. 51-76, 1988.

[12] J. J. Little and A. Verri, "Analysis of Differential and Matching Methods for Optical Flow". *IEEE Workshop on Visual Motion*, Irvine, CA, 1989.

[13] H. C. Longuet-Higgins and K. Prazdny, "The Interpretation of a Moving Retinal Image," Proc. Roy. Soc. London, ser. B. vol. 208, 1980.

[14] B. Lucas and T. Kanade, "An Iterative Image Registration Technique with an Application to Stereo Vision". *Proc. DARPA Image Understanding Workshop*, 1981.

[15] L. Matthies, R. Szelisky, and T Kanada, "Kalman Filter Based Algorithms for Estimating Depth from Image Sequences". *International Journal of Computer Vision*, vol. 2, pp. 209-237, 1989.

[16] P. S. Maybeck, *Stochastic Models, Estimation and Controls*. New York, NY: Academic Press, 1979.

[17] H. H. Nagel, "Displacement Vectors Derived from Second-Order Intensity Variations in Image Sequences". *Computer Graphic Image Process*, vol. 21, pp. 85-117, 1983.

[18] S. Negahdaripour and S. Lee, "Motion Recovery from Image Sequences Using Only First Order Optical Flow Information". *International Journal of Computer Vision*, vol. 9, pp. 163-184, 1992.

[19] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in C*: Cambridge University Press, 1992.

[20] A. Singh, *Optical Flow Computation: A Unified Perspective*: IEEE Computer Society Press, 1992.

[21] P. N. Smith, *NASA Image Data Base User's Guide*. (Vol. Version 1.0). Moffett Field, CA: NASA Ames Research Center, 1990.

[22] B. Sridhar and G. B. Chatterji, "Vision-Based Obstacle Detection and Grouping for Helicopter Guidance". *Journal of Guidance, Control and Dynamics*, vol. 17, no. 5, pp. 908-915, 1994.

[23] J. J. Wu, R. E. Rink, T. M. Caelli, and V. G. Gourishankar, "Recovery of the 3-D Location and Motion of a Rigid Object Through Camera Image (An Extended Kalman Filter Approach)". *Int. Journal of Computer Vision*, vol. 3, pp. 373-394, 1989.

[24] G. L. Zacharias, A. K. Caglayan, and J. B. Sinacori, "A Model for Visual Flow-Field Cueing and Self-Motion Estimation". *IEEE Trans on Systems, Man, & Cybernetics*, vol. 15, no. 3, pp. 385-389, 1985.

[25] G. L. Zacharias and W. H. Levison, *Development of a Model for the Use of Extra-Cockpit Visual Cues* (4562): Bolt Beranek and Newman Inc. (December), 1980.

[26] G. L. Zacharias, A. X. Miao and R. Warren "Multistage Integration Model for Human Perception". *Journal of Guidance, Control and Dynamics*, vol. 18, no. 5, pp. 937-944, 1995.

ADDENDUM

AIAA-96-3526-CP

RTPS TELEMETRY-SIMULATOR LINK AT NAVAL AIR WARFARE CENTER

William G. McNamara
Jay Nichols
Page Stanley
Naval Air Warfare Center Aircraft Division
Patuxent River, Maryland 20670

Abstract

Over the last 3 years the Naval Air Warfare Center Aircraft Division (NAWCAD), Patuxent River, MD, has been developing a link between its secure Manned Flight Simulator (MFS) and Real Time Processing System (RTPS) facilities. The MFS hosts a wide variety of high fidelity fixed and rotary wing aircraft simulation models. The RTPS is used as a telemetry ground station for conduct of Navy flight testing at Patuxent River MD. The ability to integrate simulation with flight testing in a real time environment provides new potential for increased flight safety, enhanced test team training, real time simulation fidelity assessments, and improved engineering analysis capability. A prototype system has been successfully designed and operated at NAWCAD in support of an F/A-18C flight test project which required simultaneous merging and display of real time and simulation data to reduce the risk of departure from controlled flight. More recently the link has been successfully used to provide Test Team Coordination Training (TTCT) for the F/A-18 E/F Engineering and Manufacturing Development (EMD) program. As currently designed the link (encryption and decryption gear in the loop) can be operated in three modes: (1) Simulation sending data to RTPS (e.g. pilot-engineer pre-first flight preparation/training scenario, (2) simulation is driven by real aircraft control surface inputs and response is compared with that of the real aircraft for simulation fidelity assessments and (3) simulation "rides along" with the real aircraft and data are extracted from the simulation which are otherwise unavailable from the aircraft (e.g. flight control law interconnect signals, control law feedback signals, aerodynamic data, propulsion model data, avionics model data, other model data etc.).

This paper discusses, design and implementation aspects of the RTPS-Simulator link. It includes a description of how the link was used to support a flight test program by providing real time critical safety of flight data to prevent departure from controlled flight. The link which was recently used to support initial flight testing on the F/A-18E/F EMD program is also discussed in detail.

Introduction

In 1973 the Naval Air Warfare Center Aircraft Division, Flight Test & Engineering Group (FTEG), formerly the Naval Air Test Center, established the Real Time Processing System (RTPS). RTPS has been used extensively to support numerous aircraft test programs at NAWCAD. At RTPS flight defined critical parameters are displayed in real time to project engineers for safety of flight, project monitoring and preliminary analysis. The data are displayed in Engineering Unit (EU) format onto CRT's, alphanumeric displays, and strip-chart recorders. With all these display devices and the ability to process high-sample-rate data, multiple data flights are flown daily. The EU data are recorded onto 9-track tape for post-flight analysis via high-speed remotely located processors. Data and plots can be processed and formatted more quickly, a wider variety of data and plot types can be formulated, and the data can be placed onto mainframe disk packs for future retrieval, time history and trend analysis. RTPS system architecture is illustrated in Figure 1

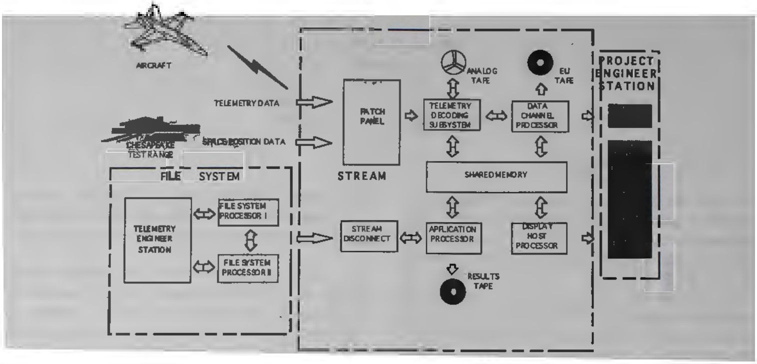


Fig 1 - RTPS Architecture

The Manned Flight Simulator Facility was established in October 1984. Since then high fidelity simulations have been implemented and used increasingly to support test and evaluation at NAWCAD. The MFS facility provides five simulation stations that support high fidelity cockpits, aerodynamic and propulsion models, and avionics simulations. The facility includes a 6 Degree of Freedom motion capable station, a fixed base 360 deg field of view 40 ft dome simulation station, two engineering development stations, and a dedicated station for the use of helmet-mounted displays. The distributed Interactive Simulation (DIS) protocol enables multiple simulation stations, covering the spectrum of air-to-air, air-to-ground, combat formation, and training scenarios, to interact with one another. Via DIS, MFS can interact with simulation facilities across the world to further test and evaluate Navy weapon systems. MFS architecture is illustrated in Figures 2 and 3.

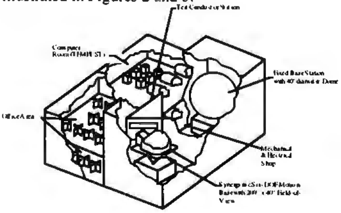

Test Conductor Station
Computer Room (TEMPEST)
Fixed Base Station with 40' diameter Dome
Office Area
Mechanical & Electrical Shop
Synergistic Six DOF Motion Base with 200 x 40' field of View

Fig 2 - MFS Facility Layout

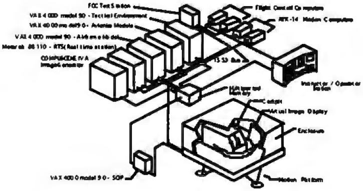


Fig 3 - MFS Architecture

While high fidelity simulations have numerous uses, a very important and frequent use at NAWCAD has been to support flight testing, particularly for reducing risk before flight. Typically, pre-test flight profiles are flown at MFS in preparation for actual flight tests. While the concept of merging simulation with flight testing is not new (e.g., X-29 real time calculation of phase and gain margin during flight envelope expansion), the use of real time simulation to support flight testing has not been extensively applied. The idea of combining simulation and real time data in a new way found support through NAWCAD in-house advanced technology initiatives. With the maturation of both the RTPS and MFS facilities at NAWCAD and recent incorporation of encryption decryption capabilities at both facilities over a T1 link the ability to link these secure facilities became a reality.

Link Description

Simulation (MFS) Link Configuration

The MFS telemetry/simulator link configuration is illustrated in Figure 4.

Fig 4 - Link Configuration

Communications are accomplished by use of a bi-directional 256,000 bits/sec rate Synchronous Data Link Communications (SDLC) interface card. The MFS/RTPS link is interfaced to the MFS through the Air Combat Environment Test and Evaluation Facility (ACETEF) Operations Control Center (OCC). The OCC is the interface for all ACETEF Laboratories of which MFS is one part. The OCC contains KG84 Encryption/Decryption equipment, a time division multiplexor-demultiplexor, and a T1 rate Network modem. The Network modem connects to the NAWCAD Patuxent River "Blue Hose" Network. This network connects all the major facilities at NAWCAD. The RTPS has a similar set of equipment to connect that facility to the network.

The RS-422 SDLC data enters MFS over a set of Fiber Optic cables. The optical data are converted by Optelecom interface equipment to RS-422 electrical signals. The RS-422 signals connect to the MFS/RTPS link processor using a Digital Equipment Corporation (DEC) DSV11 SDLC wide area network interface card. This card and its associated driver allow standard DEC VMS operating system Input/Output (I/O) calls to be made by the interface processor software. The purpose of the interface processor software is to convert the data sent by the RTPS from Encore SEL floating point format to DEC VAX floating point format and place it into the proper MFS multiport shared memory simulation area. It also takes the appropriate data from the simulation shared data area and sends it to RTPS over the SDLC link. The rate and number of variables passed is limited by the 256,000 bits/sec rate of the SDLC communications. Currently 100 32 bit variables are passed in each direction at 10 Hz. This rate and number of variables were selected to support the requirements of the first user of the link. Higher rates up to 30 samples per sec are possible with the current link configuration. Synchronization of the link is accomplished using a simple command/response method. The RTPS sends its data and in response the MFS VAX interface processor sends its data back to RTPS. The data are synchronized with a 0.1 sec time delay before simultaneous display at the Project Engineering Station.

Figure 3 (previous page) shows the distribution of the link data to the various parts of the simulation utilizing the MFS multiport memory. The multiport memory acts to tightly couple the processors that constitute the simulation into a multi-processor simulation computer system. The simulation cockpit can be moved to and interfaced with any of the MFS cockpit stations: the 40 foot DOME, the six degree of freedom Motion Base, either of the two lower fidelity display systems within the MFS computer room, or the high resolution CAE Helmet Mounted Display system.

RTPS Telemetry Link Configuration

The physical link consists of interface cards on the computers at each facility and encryption devices along with modems to send the data via the "blue hose". Each RTPS stream (6 in all) consists of three Gould computers. A Data Channel Processor (DCP) , a Display Host Processor (DHP) and an Applications Processor (APP). The APP is the processor that hosts the software that implements the link. This machine is the Gould 3297. The current capability is to send 100 parameters in each direction simultaneously at 10 samples per sec. The data rate can be increased to meet specific user requirements. The software consists of a Flight Conditions Program (FCP) that collects telemetered samples from the telemetry front end and converts the data to VAX floating point format and sends a buffer of 100 samples plus time through the link to the MFS. There is an additional "task" that will read the port from the MFS. This task will read the 100 sample buffer and place the simulation data in the Current Value Table (CVT) at RTPS in the same manner as real aircraft calculated data. These samples have already been converted to Gould Floating Point format at the other end of the link prior to transmission.

Data Flow

Data flow is as follows: Aircraft data are collected on board the aircraft and telemetered to RTPS where it is decomutated and converted to engineering units in the Aydin System 2000. Engineering units data are then transferred to a shared memory area called the Current Value Table (CVT) where it is accessible by all processors. The APP hosts the software which collects a pre-selected subset of 100 parameters, converts the parameters to VAX floating point format, and passes them to a Gould T1L1 board which converts the data to a Synchronous Data Link Card (SDLC) format at 256 kb/sec. This data stream is passed through data encryption equipment to the base wide data communications system (Blue Hose) to the MFS where it is decrypted. At this point the data may be used to send aircraft control and other inputs to a simulation software package. The outputs from this package may then be buffered and sent back through the same path (it is a full duplex system) to the RTPS. An additional software package running on the APP will read this data from the MFS and output it to the CVT as if it were any other sample of calculated data. At this point any data in the CVT may be output to any display device in the Project Engineering Station (PES). This provides a capability to simultaneously display aircraft and simulation data for direct comparison. If strip charts are used as display devices, any of the 80 pens can be selected to display either aircraft data, locally calculated data or simulator output data. CRT plots may be chosen and selected measurements or critical aircraft data and simulator data may be viewed for potential to exceed engine or aircraft pre-established limits.

Link Applications

Aircraft Departure Prevention

The link has the potential for numerous applications, one of which is real time support of flight test to provide additional safety of flight information which would otherwise be unavailable from telemetry alone. The project chosen to demonstrate this application was an F/A-18C asymmetric stores flying qualities program, completed in April 1995. The objective of this program was to expand the flight envelope and determine pilot flight restrictions with extremely large lateral weight asymmetry. A critical requirement was to prevent departure from controlled flight. Prevention of departure was mandatory because of the potential for structural damage to pylons/wing or entry into a spin from which recovery would be very difficult or not possible. The F/A-18C simulation hosted at the MFS was modified to provide critical departure prevention parameters from the simulation to be displayed to the flight test conductor.

Simulation Program Descriptions

General

For this particular application the link was configured to have the ground based F/A-18C simulation "ride along" with the real airplane. The simulation was modified such that instantaneous maximum rolling surface available deflections and corresponding rolling moment available from rolling surface versus the moment required to maintain roll control were calculated and displayed to the test conductor at RTPS in real time. If either of these parameters exceeded 75% of maximum available the test maneuver was terminated immediately to prevent departure from controlled flight. This was a unique application of simulation data because of the complex digital flight control laws which continually changed the maximum control surface deflections and rolling moments available as a function of flight conditions (i.e.; Mach number, angle of attack, normal acceleration, altitude and airspeed).

F/A-18C Simulation Modifications

The generic aircraft simulation architecture used at the MFS was modified by adding a routine that sends and receives data through global memory to the link I/O software hosted at the MFS. Two buffers are stored in this global memory, one for receiving RTPS data and the other for sending simulation data to the RTPS stream. The simulation user can interactively specify what variables are located in each of these buffer slots in the MFS memory (although the locations are hard-coded on the RTPS side), and may apply a bias and scale factor to each variable. By default (i.e., simulation rides along with real airplane) the input variables from RTPS over-write the variable values used in the simulation, such that the simulated aircraft is always at the same flight condition and configuration as the test aircraft. The output variables to RTPS are typically calculated parameters that are unavailable to the RTPS stream, such as engine forces or aerodynamic coefficients, although any simulation variable may be sent to RTPS. This can be an efficient and non-impactive way of enhancing the data available to the flight test team. The aircraft simulations hosted at MFS have already been developed, so there is no need to add and verify special calculations to the flight conditions program hosted at RTPS for each flight test program. The F-18 simulation hosted under this generic simulation architecture was modified to support the asymmetric stores loading flight test program. Two modules were added to provide two important pieces of information that are not normally available to the RTPS engineer test stations (ETS): maximum available surface deflections and maximum available roll power at a given flight condition. In order to enhance the effectiveness of this information, both were scaled from 0 to 1 for graphic display at the ETS. The following section describes the mechanization of the calculations for the parameters. An F/A-18C carrying a heavy external store on one wing only is illustrated in Figure 5.

Fig 5 - F/A-18C Single GBU-24B/B External Store Loading

F/A-18C Available Vs Used Rolling Surface Control Deflection

The MFS was asked to provide the maximum differential surface deflections available at a given flight condition so that the flight test engineers would be able to determine a "knock-it-off" point when the rolling surfaces neared saturation limits. The Project Engineer Station (PES) real time graphic display is driven by the summation of the current (i.e., actual aircraft surface positions) divided by the summation of the maximum theoretical positions corresponding to full lateral stick. The maximum theoretical positions are calculated by sending full left and right stick to a modified version of the F/A-18C lateral axis control system. The FORTRAN control laws were modified to remove dynamic filters so that rapidly changing maximum control surface limits could be determined immediately. Using this mechanization there was no delay in calculating and displaying maximum rolling surface deflection data to the test conductor. In the equation below, all entries are total differential deflections, not deltas to the average deflection. The % Rolling_Surface$_{used}$ parameter is calculated for both full left and right lateral stick deflection. The output parameter to RTPS is the ratio of current and maximum available left and right rolling surface deflection.

$$\%Rolling\_Surface_{used} = \frac{\delta_{ailerons} + \delta_{dif\_stab} + \delta_{dif\_lef} + \delta_{dif\_tef}}{\delta_{max\_ailerons} + \delta_{max\_dif\_stab} + \delta_{max\_dif\_lef} + \delta_{max\_dif\_tef}} \times 100$$

The numerator and represent the current and maximum available rolling surface deflections respectively.

F/A-18C Available Vs Used Rolling Moment

The maximum roll power available is calculated by sending the maximum surface deflections from the maximum rolling surface available calculations to a modified section of the aerodynamic model, which calculates the roll moment coefficient based on these deflections. The lateral asymmetry is calculated at RTPS and sent to the F/A-18C simulation as the rolling moment caused by the asymmetric loading at 1G. This asymmetry is multiplied by the filtered signal from the real airplane normal accelerometer. The resulting product represents the roll power that the rolling surfaces must generate to maintain roll control. The effective asymmetric roll moment shown in below is divided by the maximum roll moment available to both the left and right:

$$\%Roll\_Moment_{used} = \frac{Roll\_Moment_{asymmetry} \times Load\_Factor}{Roll\_Moment_{max\_avail}} \times 100$$

The numerator and denominator of this equation represent the current and maximum available rolling moments respectively. The ratio is an estimate of the percentage of roll power (i.e., rolling moment) used to control the asymmetric load; it implicitly shows how much roll power remains for maneuvering.

A real time display, Figure 6 was developed which utilized the F/A-18C simulation derived rolling surface and rolling moment calculations to present immediate information to the test conductor on how close the airplane was to potential departure from controlled flight. Test conductor pilot maneuver recovery calls were made when either 2/3 lateral stick or 75% rolling surface or rolling moment was needed to maintain wings level.

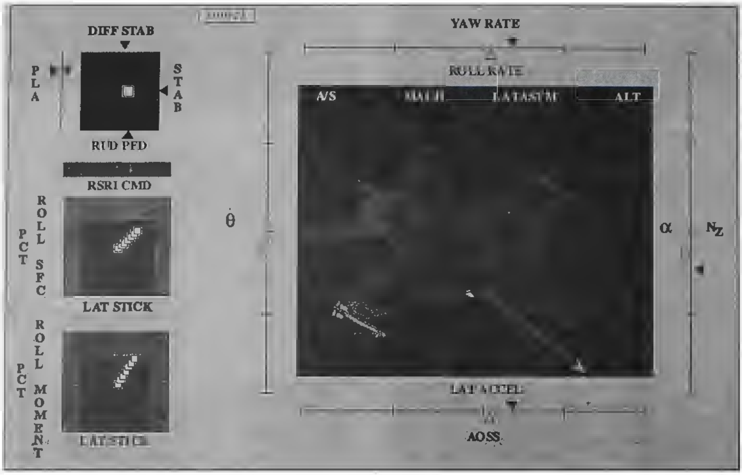


Fig 6 - Real Time Departure Prevention Display

This display was the first of its kind at NAWCAD to merge simulation and aircraft data to be used to reduce the risk of departure from controlled flight in a real time environment.

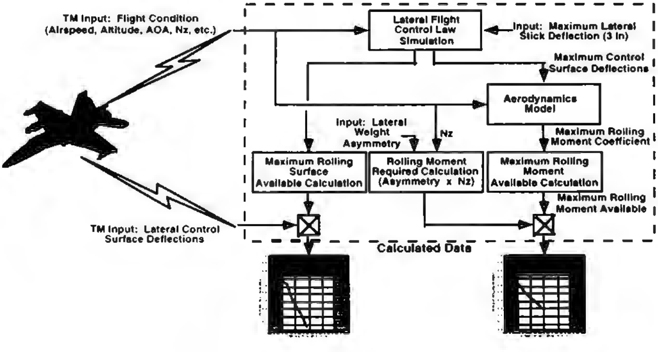

TM Input: Flight Condition (Airspeed, Altitude, AOA, Nz, etc.)
Lateral Flight Control Law Simulation
Input: Maximum Lateral Stick Deflection (3 In)
Maximum Control Surface Deflections
Aerodynamics Model
Input: Lateral Weight Asymmetry
Nz
Maximum Rolling Moment Coefficient
Maximum Rolling Surface Available Calculation
Rolling Moment Required Calculation (Asymmetry x Nz)
Maximum Rolling Moment Available Calculation
Maximum Rolling Moment Available
TM Input: Lateral Control Surface Deflections
Calculated Data

Figure 7 - Real Time Rolling Surface and Moment Calculation Process

A simplified schematic of the process used to provide real time display of maximum rolling surface and rolling moment availahle vs. used shown in Fig 7 above.

Simulation Link Test Team Coordination Training

The NAWCAD Aviation Safety Office has recognized the need to focus on "Test Team Coordination Training - TTCT" for tactical jet aircraft through the use of high fidelity simulation to enhance flight safety. The Aviation Safety Institute (ASI) conducted its ninth aviation safety evaluation of NATC (Naval Air Test Center)/NAWCAD in Jun 1995. It found that that the MFS offered significant benefits for reducing risk potential in a wide range of flight test activities. It specifically recommended that the MFS be used to mitigate risk during the F/A-18E/F EMD program^2.

TTCT encompasses test pilots flying a tactical aircraft simulation using the test site visual data base, aircraft ground controllers and test conductors and engineers at the telemetry ground station. Mission scenarios are set up at the simulation facility to evaluate, and correct as appropriate, pilot actions/reactions to various hazardous high stress aircraft system failure situations. This includes communications failures (e.g., loss of telemetry), lose of radar coverage, control system failure, engine failure (s), engine fire, massive fuel leaks etc.

F/A-18 E/F Test Team Coordination Training

During February through March 1996 the MFS-RTPS simulation link at Pax River was used for training key pilot and engineering personnel on the F/A-18E/F Integrated Test Team (ITT) in preparation for initial test flights at Pax River on aircraft "E1" and "E2". A desire to train MDA F/A-18 E/F test conductors, not familiar with Pax River flight test operations, presented an ideal opportunity to use the MFS facility high fidelity F/A-18E/F simulator and RTPS/CTR to run through initial flight test mission scenarios scheduled for aircraft "E1" and "E2". This was the first mission planning exercise of its kind at NAWCAD. The mission profile developed by the E/F ITT is illustrated in Fig 8.

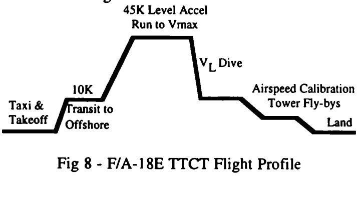


Fig 8 - F/A-18E TTCT Flight Profile

TTCT Mission Planning Description

Simulation/Cockpit Preparations

To prepare for TTCT sessions MFS was asked by the F/A-18E/F ITT to modify the simulation such that engine fires, fuel transfer failures, and "massive" fuel leaks could be simulated. Prior to formal TTCT sessions the ITT functionally checked the F/A-18E/F simulation cockpit switches and displays.

RTPS/CRT Preparations

In order to provide realistic training for the MDA test conductor and strip chart monitors, key simulation parameters were sent to RTPS to drive strip charts and various displays. The displays and strip charts were identical to that used for actual flight testing. Simulated F/A-18E/F x, y, and z coordinate information was sent to the Chesapeake Test Range (CTR) and validated against the Patuxent River simulation visual database. All data were transferred to RTPS and CTRB tracking station at 10 samples per second. The MFS fixed base 40 ft dome station was used which included a high resolution insert in the forward field of view.

Formal TTCT Simulation Sessions

For the formal TTCT simulation sessions, a massive fuel leak failure called a "Master Fuel Leak" was selected by the "E/F" ITT while the airplane was approximately 50 mi. from Patuxent River during the offshore segment of the mission profile. In this scenario, the aircraft experienced a massive loss of fuel (2000 lb/min) from the feed tank to one of the engines (tank 2 for the left engine, tank 3 for the right). Soon after the failure was inserted, the "chase plane" notified the F/A-18E test pilot that vapor was venting from the aircraft underbelly. After about twenty seconds, the pilot received a "BOOST LO" caution on his Digital Display Indicator (DDI) (indicating low pressure at the boost pump for the affected engine feed tank). If nothing were done about this situation, the affected tank would drain completely and a valve would open between the two feed tanks, causing the opposite tank to drain as well. The fuel flow was sufficient to keep both engines running, but fuel was being vented at an alarming rate. The solution (from the NATOPS) is to shutdown the affected engine, ensure the fuel leak has stopped, and proceed to an emergency landing.

TTCT Simulation Effectiveness

During TTCT simulations, the test conductors correctly identified the problem and directed the pilot in the NATOPS procedures. They then requested rigging of emergency arresting gear mid-field and instructed the pilot to return to base for an arrested landing.

The ITT indicated that using the F/A-18E simulation flight test planning mode, while not perfect, was quite effective and did provide value added. Obtaining realistic communication fidelity was challenging. The requirement to have the test pilot on "hot mike" with the test conductor at RTPS, while masking out extraneous audio from the simulation facility was difficult and will require additional work to achieve a realistic communication environment. Not unexpectedly, achieving representative audio communication fidelity was extremely important for a productive training session. A lesson learned was that it was important to have all personnel communication hooked up before performing final communication checks. Discussions with ground controllers indicted that the simulation was of extremely high fidelity with respect to ground controller pilot communications and aircraft tracking.

Lesson Learned

It is important to note that it was not originally planned to support F/A-18E/F Test Team Coordination Training. The capability to link the MFS simulation with all elements of the test team did not exist until just recently at NAWCAD. To facilitate TTCT efforts on new programs which require extensive integration and coordination between contractor and government facilities , it is highly recommended that a contractual requirement be made which integrates TTCT contractor - government simulator support into the program. This would ensure a fully supported government simulation and up front integration with ITT planning which is critical to the success of TTCT.

Future Applications

The MFS-RTPS link at NAWCAD is a new capability derived from existing capabilities looking for new and unique applications. The link can be configured in various ways for particular applications.

Other potential applications of the link include but not limited to:

Real Time Simulation Fidelity Assessment

Quick-look real time assessments of simulation fidelity by the calculation and comparison of actual and simulation predicted force and moment data for the aircraft under test. In this scenario the simulation is "driven" by the same control surface inputs as the real airplane under test. The force and moment coefficient information could be displayed in real time or post flight for determination of places within the flight envelope where the simulation exhibited good or poor fidelity. This has the potential to reduce the time it takes to update simulations to keep pace with flight testing.

Real Time Flight Control Law Analysis

For highly complex digital flight control systems not all of the flight control law signals are monitored by the aircraft multiplex bus. The ability of the simulation to provide all of the signals has the potential to significantly increase test engineer understanding of how the flight control laws are interacting with aircraft dynamics in real time. This concept is easily extended to various aircraft subsystems models which include propulsion system models, avionics models, INS models, air data system models etc. In the near future NAWCAD will have significantly increased high performance computer capabilities which enable it us to model complex systems including radar systems with high fidelity in real time.

Enhanced Ground Testing

In this mode signals form the simulation are ported to the actual airplane during ground testing and used to stimulate aircraft systems such as the flight control and hydraulic systems in a simulated airborne environment. An objective in this scenario would be enhanced understanding of system operation and risk reduction prior to actual flight tests.

Conclusions

The linking of high fidelity simulation models with real aircraft parameters in real time to enhance flight safety has been successfully demonstrated. Numerous applications have yet to be developed. As the cost of

flight testing highly complex systems increases the use of real time simulation support to reduce overall cost and enhance flight safety from a coordination resource management and training perspective will increase. Coordination resource training in the past has focused on aircrew only. In future testing environments high fidelity simulation links and high performance computers will provide the enabling technologies to expand this concept to include the entire test team. The ability to link simulation and actual aircraft data in real time has been successfully demonstrated on F/A-18C and F/A-18E T&E programs. This represents a critical validation of the potential for
high fidelity simulation to significantly improve our ability to test and evaluate complex aircraft systems in a real time environment.

Acknowledgment

The design, development and implementation of the MFS-RTPS Simulation Link over the past three years is the product of contributions by many individuals at NAWCAD among which are included: Walt Kahle, Dave Perdue, Al Conrad, Al Morrissette, and Ken Goad. The assistance of Joe Lloyd in furnishing descriptive information for this paper on RTPS capabilities is also acknowledged. Special thanks is given to Greg Dungan and Chris Clark for their enthusiasm and support for allowing the prototype link to be utilized during their asymmetric stores flying qualities flight test program. The TTCT simulation exercise involved extensive coordinated efforts of numerous engineers and pilots form the F/A-18E/F simulation team, F/A-18E/F ITT, RTPS, CTR and the Aviation System Office. Special acknowledgment is given to Capt Raymond Dudderar USN, Joe Orr, and Cdr Gordon Googe for carrying this effort forward.

References

1. McNamara, W., Stanley, P., Nichols, J. "RTPS Telemetry - Simulator Link at the Naval Air Warfare Center"; International Telemetry Conference Proceedings, Las Vegas, 1995

2. Evaluation: NATC/NAWCAD, 1995 Aviation Safety Evaluation, Aviation Safety Institute Sep 1995

Author Index

Author Index

AIAA

For permission to copy or republish, contact:
American Institute of Aeronautics and Astronautics
1801 Alexander Bell Drive
Suite 500
Reston, VA 22091

ISBN
1-56347-216-3